PolyPublie
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	Un modèle de planification tactique avec chevauchement
Auteur: Author:	Georges Baydoun
Date:	2014
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Baydoun, G. (2014). Un modèle de planification tactique avec chevauchement [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/1574/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/1574/
Directeurs de recherche: Advisors:	Robert Pellerin, & Alain Haït
Programme: Program:	Génie industriel

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

UNIVERSITÉ DE MONTRÉAL

UN MODÈLE DE PLANIFICATION TACTIQUE AVEC CHEVAUCHEMENT

GEORGES BAYDOUN
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INDUSTRIEL)
DÉCEMBRE 2014

© Georges Baydoun, 2014.

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

UN MODÈLE DE PLANIFICATION TACTIQUE AVEC CHEVAUCHEMENT

présenté par : BAYDOUN Georges
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de :

M. GAMACHE Michel, Ph. D., président
M. PELLERIN Robert, Ph. D., membre et directeur de recherche
M. HAIT Alain, Doctorat, membre et codirecteur de recherche
M. HAJJI Adnène, Ph. D., membre

REMERCIEMENTS

Mon travail de recherche est loin d'avoir été un travail individuel. Je tiens donc à remercier toutes les personnes qui, de près ou de loin, ont participé à son bon déroulement.

J'adresse d'abord mes chaleureux remerciements à mes directeurs de recherche, M. Robert Pellerin et M. Alain Haït, pour la confiance qu'ils m'ont accordée en dirigeant mon travail de recherche. Grâce à eux, j'ai bénéficié d'un encadrement et d'un support de haute qualité tout le long de mon parcours à Polytechnique. Je suis très reconnaissant des conseils judicieux qu'ils m'ont fournis, ainsi que de leurs disponibilités et grandes qualités humaines.

J'exprime également ma reconnaissance au professeur Clément Bernard pour sa précieuse aide dans l'analyse de données statistiques.

Je remercie également les membres du jury, qui me font l'honneur de juger ce mémoire et j'en profite pour leur adresser mes sincères respects.

Le cadre de travail offert par la chaire de recherche SNC-Lavalin/Jarislowsky a été fort apprécié. Je souhaiterais exprimer ma gratitude à Nathalie Perrier et François Berthaut, qui ont manifesté un intérêt pour ma recherche et qui n'ont pas hésité à diverses occasions de m'assister avec leurs expériences et expertises. Je remercie également Kaouthar Cherkaoui, avec qui les échanges ont été très bénéfiques, ainsi que Guillaume Bouvignies, qui, lors de son stage au sein de la chaire, a fourni une aide considérable à ma recherche.

Ces remerciements seraient incomplets si je ne mentionne pas mes amis, avec qui j'ai passé de beaux moments au cours de ces deux années de maîtrise. Grâce à eux, mon aventure montréalaise a été fort agréable. Je pense notamment aux deux Romain, Nicolas, Sonia et à tous mes amis qui sauront - j'en suis sûr - se reconnaître.

Finalement, un grand merci à mes parents et à ma sœur, ainsi qu'à l'ensemble de ma famille, qui m'ont encouragé à poursuivre mes études et qui m'ont toujours fourni un soutien affectif sans faille.

RÉSUMÉ

La gestion de projets est une discipline complexe, qui requiert des outils efficaces pour aider les gestionnaires dans la prise de décision. En construction, cette discipline emploie des techniques de *fast tracking*, similaires à celles de l'ingénierie simultanée, afin de réaliser les projets en régime accéléré. Une des techniques couramment employées est le chevauchement des lots de travaux, consistant à exécuter en parallèle deux lots normalement séquentiels, en commençant le lot aval tout en se basant sur des informations préliminaires du lot amont. Des outils de planification de projet avec chevauchement existent mais concernent uniquement le niveau opérationnel. À notre connaissance, les praticiens ne disposent pas d'outils les aidant dans leur choix de chevauchement lors de la planification tactique.

Compte tenu de l'importance de ces pratiques dans le milieu industriel d'aujourd'hui, ce mémoire traite de la planification tactique de projet avec des possibilités de chevauchement entre les lots de travaux. Un modèle de planification agrégée, connue aussi sous le nom de Rough-Cut Capacity Planning, avec chevauchement est proposé sous forme de programme linéaire en nombres entiers. Ce modèle combine des représentations discrète et continue du temps et considère plusieurs modes de chevauchement possibles pour les lots.

Le modèle a été testé sur 5 séries de 450 instances chacune et permet de résoudre plus de la moitié en moins de 500 secondes. Moins que le tiers de ces instances n'a pas été résolu à l'optimum dans un délai inférieur à 10000 secondes. Les résultats montrent que le chevauchement, initialement une technique d'accélération de projet, permet également la réduction des coûts en autorisant une meilleure distribution des charges.

Cependant, certaines limitations ont été soulevées à notre modèle, concernant surtout son applicabilité à la pratique. En particulier, le modèle requiert des entrées qui peuvent être difficilement accessibles pour un projet réel et la résolution exacte nécessite de long temps de calcul pour des projets de grande ampleur. Afin de contourner la deuxième limite, une alternative a été envisagée en construisant une heuristique basée sur le modèle exact. Cette heuristique semble donner des résultats encourageants. D'autres limitations ont également été soulignées concernant le caractère aléatoire des données, le type des liens d'antériorité et les fonctions objectifs considérées.

Ces limitations, ainsi que les résultats prometteurs obtenus par la solution alternative, ouvrent plusieurs voies de recherche pertinentes. Les principales avenues sont le développement d'une heuristique basée sur le modèle exact, la prise en compte de l'incertitude et la généralisation des liens d'antériorité.

ABSTRACT

Project management is a complex discipline that requires strong tools in order to help the managers make the best decisions. In construction, fast tracking techniques, similar to Concurrent Engineering techniques, are commonly employed in order to accelerate project execution. One of these techniques is the overlapping of work packages, consisting in executing in parallel two normally sequential work packages, by starting the successor before the end of the predecessor based on preliminary information. While some planning tools with overlapping exist for the operational level, to our knowledge, no such tools are dedicated for the tactical level. Thus, practitioners do not have a tool that helps them make overlapping decisions at a tactical planning.

Motivated by a true need, this study focuses on tactical project planning with the possibility of overlapping for work packages. A mixed integer linear programming model for the Rough-Cut Capacity Planning problem with overlapping is proposed. The model combines continuous and discrete representations of time, and considers several possible overlapping modes for work packages.

The model was tested on 5 sets of 450 instances each, and manages to solve more than half of instances within 500 seconds, while less than one third of the instances were not solved to optimality when reaching the time limit of 10000 seconds. Results show that overlapping can reduce project cost by allowing a better distribution of workload, even though it is primarily a technique for accelerating projects.

The model has several limitations mostly concerning its fitness to real projects applications. In fact, the model requires inputs that can be challenging to obtain in real projects, and it requires long solving times for big projects. In order to address that second limitation, a heuristic based on the exact model was proposed as an alternative solution, and gives very encouraging preliminary results. Other limitations were also raised concerning the stochastic nature of the problem, the type of precedence relations, and the considered objectives.

These limitations, as well as the promising results of the alternative solution, offer interesting prospects for the future. Possible research avenues include development of heuristics based on the exact model, consideration of uncertainty, and generalization of precedence relations.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

FEL	Front-End Loading
FLRU	Flexible Resource Loading Under Uncertainty
IAGC	Ingénierie, Approvisionnement et Gestion de la Construction
RCCP	Rough-Cut Capacity Planning
RCPSP	Resource-Constrained Project Scheduling Problem
RCPSVP	Resource-Constrained Project Scheduling Problem with Variable Intensity Activities
WP	Work Package
WPs	Work Packages

CHAPITRE 1 INTRODUCTION

La recherche dans le domaine de la planification au niveau tactique s'est traditionnellement concentrée sur des liens d'antériorité de type Fin-Début. Ces liens d'antériorité ne sont pas convenables dans un environnement où des techniques d'ingénierie parallèle, telles que le chevauchement (*Overlapping*), sont employées. Le chevauchement des activités est une pratique commune en ingénierie simultanée et en exécution de projet accéléré (*fast tracking*) qui consiste à exécuter en parallèle deux lots de travaux normalement séquentiels, en commençant le lot aval avant la fin du lot amont tout en se basant sur des informations préliminaires. Cette technique permet d'accélérer l'exécution de grands projets d'ingénierie, mais comprend des risques et des coûts représentés par les retouches (*Rework*) (Kerzner, 2001).

Ce chapitre introduira les concepts de planification agrégée et d'ingénierie simultanée, exposera les objectifs de la recherche et présentera le plan du mémoire.

1.1 Définitions et concepts de base

1.1.1 Planification agrégée

La planification agrégée d'un projet est un processus de planification avec pour principal but d'évaluer les besoins en ressources. Dans une planification agrégée, contrairement à la planification détaillée, chaque activité n'est pas différenciée, mais fait partie d'un plus gros ensemble appelé « lot de travail ». La planification agrégée est employée dans un contexte de planification hiérarchique ou de planification multiprojet.

Plusieurs structures existent pour la planification hiérarchique, incluant celle d'Anthony (1965) (planification opérationnelle, tactique et stratégique). Il existe également d'autres structures hiérarchiques, comme celle du *Front-End Loading* (FEL), décrite par Cherkaoui et al. (2013) pour les projets de construction. Dans de telles structures, une planification agrégée est préconisée avant de passer au niveau le plus détaillé, à cause des incertitudes, du manque d'information et par souci de simplification du problème. Un tel niveau d'agrégation, utilisé à un niveau tactique, a d'ailleurs été décrit par Hans (2001) dans un contexte de gestion de projet comme étant le problème *Rough-Cut Capacity Planning* (RCCP).

Par ailleurs, la planification agrégée peut être utile dans un cadre de planification multiprojet. En effet, les recherches dans le domaine de planification de projet se sont traditionnellement concentrées sur des projets isolés. Toutefois, un grand nombre de compagnies exécutent plusieurs projets en parallèle, tout en partageant les ressources entre les différents projets en

cours. Dans un tel cadre multiprojet, la planification qui considère le niveau le plus détaillé pour chaque projet devient un problème très complexe. Un niveau agrégé pour chaque projet est alors plus adéquat, car il facilite l'analyse et la communication (Kerzner, 2001; Talla et al., 2014).

Dans le but d'accélérer l'exécution de projets, la gestion de projet emploie des techniques d'ingénierie simultanée, initialement développées dans le cadre du développement de produits. Le paragraphe suivant introduira cette méthodologie.

1.1.2 L'ingénierie simultanée

L'ingénierie simultanée (*Concurrent Engineering*) est une méthodologie de développement de produits qui s'est développée depuis les années 1980 dans le but de réduire les temps de mise en marché de nouveaux produits. Elle est considérée comme étant l'initiative ayant le plus influé le développement de produits dans les années 1990 (Backhouse and Brookes, 1996). Initialement développée dans un contexte manufacturier, elle a également intéressé le domaine de la construction (Bogus et al., 2005).

L'ingénierie simultanée s'est développée dans un contexte où les marchés deviennent plus exigeants en terme de qualité et coûts, imposant aux entreprises leurs propres règles. Les développements technologiques et l'apparition permanente de nouveaux produits ont considérablement réduit les temps de cycle, rendant les produits plus rapidement obsolètes (Backhouse and Brookes, 1996). C'est pourquoi la diminution de la durée de mise sur le marché devient un véritable avantage concurrentiel pour les entreprises.

L'ingénierie simultanée consiste à effectuer plusieurs phases en parallèle s'opposant ainsi à l'ingénierie séquentielle où chaque étape démarre quand la précédente se termine. Suivant cette méthodologie, tous les éléments tels que le marketing, l'ingénierie, la production et l'approvisionnement sont pris en compte simultanément dès le départ (Koufteros et al., 2001). Il n'existe pas une définition unique pour l'ingénierie simultanée, mais celle qui est communément utilisée est celle qui est proposée par Winner et al. (1988) :

> *" Concurrent Engineering is a systematic approach to the integrated, concurrent design of products and their related processes, including, manufacturing and support. This approach is intended to cause the developers from the very outset to consider all elements of the product life cycle, from conception to disposal, including quality, cost, schedule, and user requirements. "*

Une technique répandue d'ingénierie simultanée est le chevauchement des lots de travaux, où un lot commence avant que son prédécesseur ne finisse. Les liens classiques d'antériorité de

type Fin-Début deviennent alors inadéquats pour cette technique.

1.2 Objectifs de recherche

L'objectif de la recherche est de proposer un modèle de planification de projet agrégée qui intègre la possibilité de chevauchement entre les lots de travaux. Pour ce faire, un modèle linéaire mixte à variables binaires et continues sera proposé. Des instances de tests seront créées et permettront de tester le modèle mathématique afin de le valider et d'analyser ses résultats et ses performances. Finalement, une heuristique destinée aux projets de grandes tailles sera proposée.

1.3 Plan du mémoire

Le mémoire s'organisera de la manière suivante. Dans le chapitre 2, une revue de littérature présente des approches existantes pour le problème du RCCP et ses extensions, ainsi que pour le chevauchement. Le chapitre comprend ensuite une revue critique de la littérature (section 2.6) qui met en avant l'absence de modèle de planification tactique autorisant le chevauchement des lots de travaux, ce qui justifie l'intérêt de proposer un modèle RCCP avec chevauchement. Dans le chapitre 3, nous développons notre méthodologie de recherche. Ensuite, nous intégrons dans le chapitre 4 un article qui présente le modèle sous forme de programme linéaire mixte à variables binaires et continues. La génération des instances de test, les analyses concernant les performances et les résultats obtenus sont également présentées dans cet article. Par la suite, les limitations de la solution proposée, les voies d'améliorations, une solution alternative et les implications pratiques du modèle sont discutés dans le chapitre 5. Dans la conclusion (chapitre 6), l'approche proposée et les résultats obtenus sont récapitulés et les contributions de la recherche identifiées, avant de terminer avec un rappel des limitations et une présentation de futures opportunités de recherche.

CHAPITRE 2 REVUE DE LA LITTÉRATURE

Le but de ce chapitre est de présenter la revue de la littérature sur le problème *Rough-Cut Capacity Planning* (RCCP) et le chevauchement. La revue de littérature se divise en sept parties. La première décrit le problème RCCP et le compare au problème *Resource-Constrained Project Scheduling Problem* (RCPSP). La deuxième partie discute de deux approches, exactes et approchées, de résolution du RCCP classique. Ensuite, différentes extensions au RCCP sont présentées dans la troisième partie. La quatrième section introduit deux structures de planification hiérarchique et positionne le problème RCCP au sein de ces structures. La cinquième partie définit le chevauchement et recense des modèles de chevauchement dans la littérature. À la fin de ce chapitre, une analyse critique de la littérature permettra de conclure que le RCCP avec chevauchement n'a pas encore été traité, d'où l'originalité et l'intérêt de proposer un modèle RCCP avec chevauchement.

2.1 Le problème Rough-cut Capacity Planning

Lors de la phase de négociations et d'acceptation d'un projet, les entreprises sont amenées à prendre des décisions d'acceptation ou de rejet, en dépit de l'incertitude et des manques d'informations (Wullink, 2005). Dès les premières phases du projet, les clients ont tendance à exiger des entreprises un engagement sur des dates et des coûts qu'elles devront respecter lors de l'exécution. C'est ainsi qu'une bonne planification devient un véritable atout pour les entreprises.

Le RCCP permet de planifier les lots de travaux et d'estimer de manière agrégée les quantités requises de ressources, ce qui permet de déterminer des dates jalon fiables et de gérer convenablement les ressources. L'approche équivalente au RCCP dans un environnement *Engineering-To-Order* (ETO) est celle du *Resource Loading* (Hans, 2001; Wullink, 2005). Utilisé à un niveau tactique, le RCCP présente plusieurs points de différence avec le problème opérationnel RCPSP, qui seront expliqués dans la partie suivante.

2.1.1 Différences entre le problème RCCP et RCPSP

Dans un contexte hiérarchique, la planification RCCP intervient plus tôt dans le temps que la planification RCPSP et concerne la planification de la capacité des ressources à moyen terme ainsi que la détermination de dates jalons fiables. Le RCPSP intervient plus tard et vise l'ordonnancement des tâches à court terme (De Boer, 1998).

Le degré de précision et de disponibilité des données est un autre point de différence entre les deux problèmes. En effet, au moment où le RCCP intervient, les informations sont incomplètes ou imprécises. C'est pourquoi les projets sont divisés en activités agrégées qui sont des regroupements de plusieurs activités. Les informations disponibles sur ces activités agrégées sont leurs consommations estimées en ressources, leurs durées minimales, ainsi que leurs liens de précédence (Cherkaoui et al., 2013). La quantité de travail requise est connue, mais les durées d'exécution (liée au mode d'exécution) ne sont pas une donnée du RCCP, contrairement au RCPSP où les durées des tâches sont connues.

Par ailleurs, les contraintes des ressources diffèrent entre le RCCP et le RCPSP. En effet, pour le RCCP, l'horizon de temps est divisé en périodes sur lesquelles la consommation des ressources est évaluée, tandis que l'horizon de temps est continu pour le RCPSP. Le RCCP permet alors de déterminer les capacités requises des ressources pour chaque période, tout en autorisant d'augmenter les capacités régulières par des capacités non régulières, plus coûteuses. Ces dernières proviennent par exemple de la sous-traitance, de l'embauche d'intérimaires ou des heures supplémentaires (Wullink, 2005). En ce sens, les contraintes de ressources du RCCP sont plus souples que celles du RCPSP où il existe très peu ou pas du tout de flexibilité dans les capacités des ressources (Gademann and Schutten, 2005).

Finalement, le RCPSP a souvent un objectif lié au temps (minimiser le retard, ou la durée du projet). Quant au RCCP, deux approches peuvent être distinguées : *Time-driven approach* et *Resource-Driven approach* (Möhring, 1984). La première minimise le coût d'emploi de ressources non régulières tout en respectant les dates d'échéances, alors que la deuxième fixe les contraintes de ressources et tente de minimiser le retard du projet. Une combinaison possible des deux approches consiste à trouver le compromis entre l'utilisation de ressources non régulières et les pénalités de dépassement des dates de fin (Wullink et al., 2004).

2.2 Méthodes de résolution du RCCP

Contrairement au problème RCPSP qui a été abordé par un grand nombre d'auteurs, la littérature sur le RCCP reste très restreinte. Nous nous intéressons dans cette partie aux méthodes de résolution RCCP ainsi qu'aux méthodes d'ordonnancement adaptées à la vision agrégée du niveau tactique. Ceci concerne plus particulièrement les modèles d'ordonnancement à intensités variables. Deux approches de résolution sont distinguées dans cette partie : les méthodes exactes et approchées.

2.2.1 Méthodes exactes

Weglarz (1981) étudie l'ordonnancement avec des consommations de ressources à intensités variables des activités, où l'objectif est de minimiser la durée du projet. Une fonction continue de l'utilisation temporelle des ressources permet de déterminer l'intensité de chaque activité. Deux types de ressources sont distingués par Weglarz : les ressources renouvelables et non renouvelables. La consommation du premier type de ressources est contrainte à tout moment, alors que celle des ressources non renouvelables est contrainte par période.

Pour sa part, Hans (2001) propose une approche exacte de planification tactique à l'aide d'un programme linéaire en nombres entiers (PLNE). Hans considère des périodes de temps sur lesquelles sont évaluées les consommations en ressources. La notion de plan de projet est centrale dans sa formulation. Un plan de projet spécifie pour chaque activité les périodes pendant lesquelles elle peut être exécutée. Les plans faisables tiennent compte des fenêtres de temps pour les activités et des liens de précédence. Le calendrier de projet vient ensuite déterminer pour chaque période les fractions des activités qui doivent être exécutées pour minimiser les coûts de l'utilisation de ressources non régulières et des pénalités de retard. Étant donné le grand nombre de plans de projet faisables, Hans propose une approche de type *Branch-and-Price* qui génère au besoin des plans de projet. Il suggère plusieurs possibilités pour traiter les liens de précédence dans ses plans. La première consiste à autoriser les activités avec des liens de précédence à être exécutées sur une même période. Cependant, ceci peut conduire à des cas où les fractions obtenues des activités ne peuvent pas être effectuées séquentiellement sur une même période de manière à respecter la précédence. Une autre possibilité est de limiter le nombre d'activités sur une même période dans les plans de projet, ce qui réduit le risque d'infaisabilités. La troisième alternative est d'interdire les activités ayant un lien de précédence direct d'être sur une même période. Cette possibilité élimine le risque d'infaisabilités en sur-contraignant le problème.

Kis (2005) propose une extension au problème RCPSP dans laquelle l'intensité des activités est variable dans le temps et l'utilisation des ressources est proportionnelle à l'intensité. Kis prouve que son problème, qu'il nomme RCPSVP, est NP-difficile au sens fort. Il propose un programme linéaire en nombres entiers dans lequel deux activités ayant une relation de précédence directe ne peuvent pas être effectuées dans une même période. Une étude polyédrique permet ensuite de déterminer un ensemble de coupes qui seront générées dans une approche de type *Branch-and-Cut*. Une telle approche est duale à l'approche *Brand-and-Price* de Hans (2001). En effet, les deux auteurs traitent le même problème de la même manière, mais avec deux modélisations différentes des contraintes de précédence. La formulation de Hans (2001) pourrait conduire à un très grand nombre de variables, d'où l'approche de

résolution par génération de colonne, alors que la formulation de Kis conduit à un très grand nombre de contraintes, d'où l'approche par génération de ligne *Brand-and-Cut*. Plus tard, Bianco and Caramia (2013) proposent une variante au modèle de Kis (2005), où les intensités des tâches sont cumulatives.

Chen et al. (2009) proposent un programme linéaire en nombres entiers pour le problème de la planification de la capacité à court terme (*Short-term Capacity Planning*) dans un environnement de fabrication à la commande (*Make-To-Order*). Le modèle divise également l'horizon de temps en périodes sur lesquelles les contraintes de capacité des ressources sont évaluées. Des variables binaires permettent de déterminer les périodes pendant lesquelles les tâches sont réalisées ainsi que le type du travail (heures régulières, heures supplémentaires et/ou sous-traitance). Dans ce modèle, la durée de traitement de chaque tâche sur une ressource est une donnée du problème, mais le nombre d'heures assignées (et donc l'intensité d'utilisation de la ressource par la tâche) durant chaque période est une variable.

Le modèle de Haït and Baydoun (2012) combine une représentation continue du temps pour le début et la fin des activités, avec une représentation discrète pour l'évaluation de la consommation en ressources. Ainsi, les liens de précédence sont exprimés simplement avec les dates de début et fin des activités, alors que les contraintes sur les ressources s'expriment par période. De plus, Haït and Baydoun définissent pour chaque activité des intensités minimales et maximales à respecter lors de l'exécution. Un ensemble de variables binaires assure le lien entre les évènements de début et fin des activités avec les durées par périodes. Ces durées permettent de retrouver les charges maximales pouvant être allouées par périodes pour respecter les intensités minimales et maximales. Une telle approche permet de contourner le problème lié aux liens de précédence soulevé par Hans (2001) et Kis (2005).

Plus récemment, Naber and Kolisch (2014) ont adressé le problème RCPSP avec des profils flexibles de ressources, signifiant que la consommation des ressources par une activité peut varier d'une période à l'autre. Les auteurs proposent plusieurs formulations sous forme de programmes linéaires mixtes basés sur des formulations existantes et comparent leurs performances. Il s'avère alors que le modèle nommé "FP-DT3", basé sur la formulation RCPSP de Bianco and Caramia (2013) et sur le modèle RCPSVP de Kis (2005) avait de meilleures performances que les autres formulations aussi bien sur le point de vue de la qualité de la solution obtenue, que sur le temps de calcul.

2.2.2 Méthodes approchées

Les problèmes RCCP et RCPSP (et leurs extensions) étant NP-difficiles au sens fort, les méthodes exactes sont incapables de résoudre de grandes instances en un temps raisonnable.

Comme le remarque Herroelen (2005), les méthodes exactes de RCPSP sont d'ailleurs peu utilisées dans la pratique pour la planification. C'est pourquoi des méthodes approchées ont été proposées dans la littérature afin de trouver rapidement des solutions non forcément optimales. L'intérêt des méthodes exactes reste de trouver les solutions optimales de certaines instances afin d'évaluer la performance des méthodes approchées, ainsi que de servir de base à certaines méthodes approchées.

De Boer (1998) propose plusieurs heuristiques pour le problème RCCP. Son heuristique ICPA (*Incremental Capacity Planning Algorithm*) est de type constructif et s'effectue en deux phases. Pour chaque activité, on commence par allouer à sa fenêtre de temps le maximum possible de la charge requise en utilisant uniquement des capacités régulières. S'il reste une charge non allouée, on utilise de la capacité non régulière. Cette heuristique est destinée au problème *time-driven* RCCP, mais peut-être utilisée pour le *resource-driven* RCCP. Une autre heuristique, basée sur une formulation en programme linéaire, est également proposée pour le problème *time-driven* RCCP. Elle consiste à résoudre d'abord un programme linéaire où les relations de précédence sont négligées et à réparer ensuite les relations de précédence violées. Les résultats obtenus avec la deuxième heuristique sont meilleurs, mais demandent un temps de calcul plus important.

Gademann and Schutten (2005) proposent également un ensemble d'heuristiques. Selon les auteurs, les relations de précédence rendent le problème difficile à résoudre, d'où l'idée de réduire le problème à un programme linéaire avec des fenêtres ATW (*Allowed-To-Work*), qui garantissent les liens de précédence. Les heuristiques proposées tentent de déterminer de bonnes fenêtres ATW. Un premier type d'heuristiques proposées est constructif. Ces heuristiques ne donnent pas de bons résultats, mais permettent de fournir une solution initiale. Cette solution peut servir d'entrant pour un deuxième type d'heuristiques proposées par les auteurs et destinées à améliorer des solutions admissibles par modification des fenêtres ATW. Finalement, les auteurs proposent une troisième catégorie d'heuristiques qui réparent des solutions non réalisables. Toutes ces heuristiques sont destinées au *time-driven* RCCP.

2.3 Extensions au problème RCCP classique

Il existe dans la littérature des variantes ou des extensions au problème RCCP déterministe classique. Les plus importantes concernent la prise en compte de l'incertitude et la généralisation des liens de précédence.

2.3.1 Prise en compte de l'incertitude

La grande majorité des modèles de planification au niveau tactique sont déterministes, ce qui est peu réaliste pour des environnements incertains. C'est pourquoi certains auteurs ont tenté de tenir compte de l'incertitude au niveau tactique.

Wullink et al. (2004) considèrent une extension au modèle de Hans (2001) qui tient en compte l'incertitude sur les durées de traitement. Ce modèle, qu'ils nomment *Flexible Resource Loading Under Uncertainty* (FRLU), a pour but de minimiser les coûts espérés. Pour cela, ils considèrent pour chaque activité trois charges requises possibles, appelés modes. Un scénario correspond alors au choix d'un mode d'exécution par activité.

La méthode exacte pour résoudre le FRLU est basée sur la méthode *Branch-and-Price* de Hans (2001). Cette méthode ne trouve généralement pas de solution en un temps raisonnable. Wullink et al. (2004) proposent alors d'appliquer l'heuristique de Gademann and Schutten basée sur la programmation linéaire afin de réduire le temps de calcul. Étant donné le grand nombre de scénarios possibles, une technique d'échantillonnage est également suggérée par les auteurs qui consiste à sélectionner un certain nombre de scénarios. L'heuristique combinée à l'échantillonnage donne des résultats satisfaisants en un temps raisonnable.

Masmoudi et al. (2014) traitent également du problème RCCP avec des charges de travail incertaines. Contrairement à Wullink et al. (2004), les distributions sont continues. L'approche utilisée est le celle de la métaheuristique du recuit simulé, appliquée au modèle de Hans (2001). Une heuristique basique permet de construire une solution initiale faisable. Une solution au sens de Hans (2001) consiste en un plan de projet et un calendrier de projet. Ensuite on perturbe cette solution initiale en modifiant soit le plan de projet, soit le calendrier de projet. Si la solution perturbée améliore la fonction objectif, on sélectionne la nouvelle solution. Sinon, on sélectionne la nouvelle solution avec une probabilité liée au facteur de Boltzmann. Ceci est répété jusqu'à l'atteinte du critère d'arrêt. La méthode est compétitive avec celle de Gademann and Schutten pour beaucoup d'instances et permet d'améliorer les résultats sur les grandes instances tout en consommant plus de temps de calcul.

2.3.2 Généralisation des contraintes de précédence

Tous les modèles cités considèrent de simples relations de précédence Fin-Début. En réalité, les liens de précédence entre les activités sont plus compliqués. Certains auteurs ont essayé de tenir compte de ce fait en introduisant de nouvelles possibilités pour les liens de précédence.

Kis (2006) introduit au modèle RCPSVP (Kis, 2005) des relations de précédence qu'il nomme *feeding precedence constraints*. Ce type de contraintes autorisant le successeur à commencer

quand un certain pourcentage (donnée du problème) du prédécesseur est complété. Si le pourcentage est strictement inférieur à 100%, cela veut dire que le successeur peut chevaucher sur le prédécesseur. Si le pourcentage est de 100%, on retrouve de simples relations Fin-Début. Kis (2006) adapte le modèle RCPSVP pour tenir compte des nouvelles relations de précédence et refait une étude polyédrique afin de trouver des contraintes spécifiques au nouveau problème. En appliquant à nouveau son approche *Branch-and-Cut* sur des instances modifiées de De Boer (1998), Kis (2006) trouve de meilleurs résultats, ce qui laisse penser l'auteur que le nouveau problème est plus facile que le RCPSVP.

Alfieri et al. (2011) reprennent la même idée de Kis (2006) et l'appliquent à des relations de précédence généralisées. La généralisation des liens d'antériorité n'est pas nouvelle en soi (Elmaghraby and Kamburowski, 1992), mais la nouveauté est son adaptation à des liens de précédence définis par les pourcentages de complétion. Alfieri et al. (2011) distinguent quatre type de relations de précédence généralisées : *%Completed-to-Start*, *Start-to-%Completed*, *%Completed-to-Finish* et *Finish-to-%Completed*. Les contraintes de type *%Completed-to-Start* correspondent aux relations définies par Kis (2006). Alfieri et al. (2011) proposent deux formulations mathématiques qu'ils testent sur des instances générées aléatoirement, ainsi que sur des cas réels en utilisant le logiciel commercial CPLEX. Bianco and Caramia (2012) reprennent les travaux de Alfieri et al. (2011) et proposent un algorithme exact. Leur algorithme arrive à résoudre de plus grandes instances que le logiciel commercial et nécessite de plus petits temps de calcul.

Nous avons vu que la planification RCCP intervient habituellement plus tôt que la planification opérationnelle et à un niveau de planification plus élevé. La section suivante positionnera le problème RCCP au sein d'une structure générique de planification hiérarchique, ainsi que dans le cadre d'une structure plus adaptée aux projets de construction.

2.4 Positionnement du problème

2.4.1 Structure générique de planification hiérarchique

Selon la typologie d'Anthony (1965), il existe trois niveaux de planification : la planification stratégique, tactique et opérationnelle. De Boer (1998) a proposé une structure hiérarchique basée sur ces trois niveaux, qui a inspiré les travaux de plusieurs auteurs.

Planification stratégique

Le niveau stratégique implique des décisions à long terme (un horizon de temps d'une à plusieurs années) prises au niveau de la haute direction des entreprises (Cherkaoui et al., 2013). Ce niveau de planification traite de problèmes stratégiques comme le recrutement à grande échelle, les changements de localisation ou de services et la gestion des ressources critiques. Des méthodes de planification linéaires sont généralement utilisées pour ce niveau de planification. La particularité de la planification stratégique est qu'elle se base sur une prévision de la demande et non sur des commandes ou des projets spécifiques de clients. Ceci rend les méthodes de planification stratégique inadaptées aux niveaux tactique et opérationnel qui considèrent des données spécifiques et non pas prévues (Wullink, 2005).

Planification tactique

Au niveau tactique, la planification aborde les problèmes d'allocation efficiente des ressources aux commandes des clients et de la détermination de dates jalons fiables. Les méthodes de planification tactique peuvent également servir d'aide à la décision lors de la phase de réponse à un appel d'offres, en évaluant la faisabilité ou la rentabilité d'un projet. À ce niveau intermédiaire, les planificateurs ne possèdent que des informations approximatives sur le contenu des commandes et sur la disponibilité des ressources régulières. Au niveau tactique, le RCCP est utilisé pour la planification en gestion de projet, avec pour équivalent dans un contexte de fabrication à la commande le problème *Resource Loading* (Hans, 2001).

Les liens existant entre les différent niveaux hiérarchiques ont été étudiés par Hans et al. (2007) en fonction de la dépendance des projets à des facteurs externes (par exemple l'approvisionnement ou les sous-traitants) et internes (par exemple des ressources partagées entre différents projets). Pour les organisations à forte dépendance, les quantités requises de ressources et les jalons sont une sortie du RCCP, servant de données d'entrée pour le niveau opérationnel. Pour les organisations à faible dépendance, l'allocation des ressources peut être une donnée connue au niveau tactique.

Planification opérationnelle

Finalement, le niveau opérationnel de planification permet l'ordonnancement des activités à court terme, avec généralement un objectif lié au temps. La capacité des ressources n'est plus flexible (Wullink, 2005). De Boer (1998) propose pour ce niveau de résoudre le problème d'ordonnancement avec contraintes de ressources (RCPSP). Le RCPSP vise à déterminer l'ordre d'exécution des activités pour minimiser la durée du projet, tout en respectant des

contraintes de ressources et de précédence entre les activités (Cherkaoui et al., 2013).

2.4.2 Planification hiérarchique dans les projets de construction

En construction, la majorité des grands projets d'ingénierie sont réalisés en engageant un contractant principal qui se charge de l'ingénierie, de l'approvisionnement et de la gestion de la construction au nom du client, selon un contrat de type IAGC (Ingénierie, Approvisionnement et Gestion de la Construction). En pratique, la planification de ces grands projets s'effectue en créant des échéanciers à différentes phases du projet. Le niveau d'agrégation dépend de la phase en cours et du public visé par le plan (Cherkaoui et al., 2013). Les auteurs expliquent la pratique adoptée pour la planification de tels projets. Trois grandes phases sont distinguées : pré exécution, exécution, mise en service et démarrage de projet. La phase de pré exécution est appelée *Front-End Loading* (FEL). Après le FEL, le contrat IAGC est alloué à un constructeur pour effectuer les phases d'exécution : ingénierie détaillée, approvisionnement et construction. Selon le *Independant Project Analysis* (IPA), la phase FEL permet à l'entreprise une définition détaillée du projet afin d'atteindre ses objectifs. Le FEL est généralement décomposé en trois phases : FEL1, FEL2 et FEL3. La première phase FEL1 permet de valider les opportunités d'affaires du projet, valider les objectifs du projet, construire une estimation des coûts (de typiquement -25/+40%) et fixer les jalons majeurs du projet. Durant la phase FEL2, plusieurs alternatives répondant aux objectifs du projet sont proposées et étudiées selon plusieurs critères (par exemple techniques, financiers et environnementaux). À la fin de cette phase, une seule alternative est retenue et des jalons importants sont fixés (tels que la date d'approbation du budget). Finalement, la stratégie retenue est développée au cours du FEL3, où des techniques telles que le RCCP sont employées afin de fixer les jalons et s'assurer que les ressources sont suffisantes pour mener à bien le projet. La planification durant les phases FEL3 correspond au niveau tactique, alors que la planification durant la phase d'exécution correspond au niveau opérationnel (Cherkaoui et al., 2013). Dans la pratique, les trois phases d'exécution peuvent se chevaucher afin d'accélérer le projet. Il s'agit de la technique de chevauchement employée dans le cadre de l'ingénierie simultanée, qui sera l'objet de la section 2.5.

2.5 La technique du chevauchement

2.5.1 Le chevauchement

Dans le but de diminuer les dates de fin des projets, l'ingénierie simultanée préconise le chevauchement (*overlapping*) des processus et des activités (Bogus et al. (2005), Prasad

(1996)). Le chevauchement consiste à exécuter simultanément deux activités normalement séquentielles, en commençant le successeur avant la fin de son prédécesseur (Grèze et al., 2012). Basée sur des informations préliminaires et incomplètes, cette technique est cependant risquée et accroît la possibilité d'erreurs et de retouches (*rework*) (Dehghan and Ruwnapura, 2014). Un exemple serait de commencer la construction d'une usine avant que les plans ne soient complètement finalisés. Un éventuel changement dans les plans aura alors pour conséquence de rajouter des travaux de modification en construction afin de respecter les nouveaux plans. Ainsi le chevauchement pourrait avoir un effet inverse au résultat désiré à cause des *reworks*, d'où l'intérêt de trouver le bon compromis entre diminuer le temps par chevauchement et rajouter du travail de retouches.

2.5.2 Modèles de chevauchement entre deux activités

Une première partie de la littérature sur le chevauchement s'est intéressée à une seule paire d'activités ou de phases chevauchables. Le flux d'information et la coordination sont centraux dans ces modèles.

Krishnan et al. (1997) étudient le chevauchement entre deux activités en considérant deux caractéristiques des activités amont et aval permettant de définir plusieurs cas de figure. La première caractéristique est le degré d'évolution de l'activité amont, qui concerne la rapidité de la finalisation de l'information issue de l'activité amont utilisée par l'activité aval. La deuxième est la sensibilité et la rapidité d'adaptation de l'activité aval vis-à-vis des changements dans l'activité amont. Le modèle mathématique permet alors de trouver le chevauchement et la stratégie de communication qui permettent de minimiser la durée totale.

Peña-Mora and Li (2001) proposent une méthode de planification dynamique appliquée aux projets de construction. Trois caractéristiques sont considérées : le taux de production, la fiabilité de la production de la tâche amont et la sensibilité aux erreurs de la tâche aval. Bogus et al. (2005) s'intéressent également au chevauchement dans les projets de construction en utilisant les mêmes caractéristiques des tâches que Krishnan et al. (1997). Ils présentent un processus de caractérisation permettant aux gestionnaires de trouver des pairs d'activités convenables au chevauchement.

Loch and Terwiesch (1998) proposent un modèle analytique qui détermine le chevauchement optimal entre deux activités séquentielles et la fréquence des échanges d'informations. Ils trouvent alors que l'incertitude et la dépendance entre les activités augmentent la quantité requise de communication, rendant le chevauchement moins attractif.

Les conclusions de Loch and Terwiesch (1998) sont en accord avec les résultats numériques de

Ha and Porteus (1995) qui présentent un programme dynamique pour la politique d'échange d'information entre deux activités qui doivent s'effectuer en parallèle. Plus le temps consacré aux échanges d'informations est grand, moins les retouches sont nécessaires. Le modèle permet alors de trouver le compromis entre la durée et la fréquence des échanges d'informations et les gains en retouches.

Lin et al. (2010) proposent un modèle analytique qui détermine le taux optimal de chevauchement et la politique de communication permettant de maximiser les performances du projet (compromis entre durée et coût). Le modèle de Lin et al. (2010) tient compte de l'effet de l'évolution de l'activité en aval sur la stratégie de chevauchement optimale.

2.5.3 Modèles de chevauchement entre plusieurs couples d'activités

Tous les auteurs cités dans la partie précédente étudient le chevauchement d'une seule paire d'activités. D'autres études considèrent plusieurs couples d'activités pouvant se chevaucher.

Nicoletti and Nicolo (1998) proposent un modèle linéaire de planification avec chevauchement. L'objectif de ce modèle est de maximiser un "index" lié au flux d'information entre les tâches simultanées, qui est en réalité une mesure du taux de chevauchement dans le projet.

Roemer and Ahmadi (2004) présentent un modèle permettant de trouver le compromis entre le coût et la durée du projet. En considérant le chevauchement et la compression (*crashing*), ils proposent une extension au modèle de Roemer et al. (2000) qui considère uniquement le chevauchement entre les phases. Le modèle de Roemer and Ahmadi (2004) considère plusieurs couples pouvant chevaucher, mais se limite aux projets constitués de phases séquentielles.

Gerk and Qassim (2008) étudient l'accélération de projets de trois manières : le chevauchement, la compression et la substitution des activités, avec contraintes de ressources. Ils modélisent le problème par un programme non linéaire en nombres entiers, qu'ils transforment en programme linéaire, ayant pour objectif la réduction des coûts d'accélération du projet. Dans le modèle de Gerk and Qassim (2008), la relation entre le taux de retouches et degré de chevauchement est linéaire.

Des études similaires récentes s'intéressent au chevauchement des activités à l'échelle de projet. Berthaut et al. (2014) proposent une extension au problème RCPSP qui autorise le chevauchement entre les activités. La particularité du modèle est l'existence d'un nombre limité de modes de chevauchement par couple. Berthaut et al. (2014) déduisent que les contraintes de ressources limitent les bénéfices du chevauchement puisqu'il engendre une consommation supplémentaire de ressources. Grèze et al. (2012) étudient le même problème que Berthaut et al. (2014) mais fournissent une heuristique constructive, adaptée à des projets

de grande taille.

Finalement, Dehghan and Ruwnapura (2014) proposent un modèle de chevauchement entre les activités dans un projet de construction. Les auteurs ont effectué une revue de littérature puis interrogé des experts afin de générer un modèle réaliste. La particularité de leur modèle est qu'il traite du cas où plusieurs chevauchements se superposent (*Cascade Effect*). L'objectif du modèle proposé est de minimiser le coût d'accélération du projet, mais les auteurs suggèrent plusieurs autres objectifs possibles. L'article n'aborde pas la résolution du modèle, mais les auteurs affirment que de futurs articles traiteront d'un algorithme développé à cette fin.

2.6 Revue critique de la littérature

Le chevauchement est une réalité dans beaucoup de projets d'ingénierie dans le but de réduire les délais. Il convient donc de tenir compte du chevauchement dans la planification dès le niveau tactique. Or la plupart des modèles du niveau tactique considèrent de simples relations Fin-Début. Certains modèles existants permettent d'effectuer des lots de travaux en parallèle, mais ne sont pas de véritables modèles de chevauchement. En effet, ces modèles considèrent uniquement une seule possibilité de lien d'antériorité entre deux lots et non pas plusieurs modes de chevauchements possibles. De plus, les charges de travail des lots y sont fixées et ne dépendent pas donc du chevauchement qui devrait engendrer des charges additionnelles de *rework*.

Par ailleurs, tous les modèles de chevauchement existants pour les projets ne sont pas adaptés au niveau tactique, car ils considèrent des durées et non pas des charges de travaux pour les tâches. Les retouches sont présentes dans ces modèles, mais concernent uniquement le successeur qui chevauche ; le travail requis pour effectuer le prédécesseur n'est pas affecté par des retouches. Or, en pratique, le chevauchement entre deux lots de travaux nécessite la multiplication de rencontres et une coordination accrue entre les deux lots. Ceci peut être modélisé par un *rework* sur le successeur et le prédécesseur qui chevauchent.

2.7 Conclusion

Dans ce chapitre, nous avons effectué une revue de littérature. Le chapitre s'est organisé selon deux axes principaux : le RCCP et la technique du chevauchement. La revue de littérature nous a permis de repérer un manque de modèle de planification adapté au niveau tactique, qui autorise le chevauchement entre les lots de travaux suivant plusieurs modes possibles.

Afin de combler cette lacune, nous proposons dans le chapitre suivant un modèle RCCP avec chevauchement. Dans notre modèle il existe plusieurs modes de chevauchement discrets pour les couples pouvant chevaucher. Chaque mode se traduit par un pourcentage du prédécesseur à partir duquel le successeur peut commencer et une quantité de retouches sur le prédécesseur et sur le successeur.

CHAPITRE 3 MÉTHODOLOGIE DE RECHERCHE

Ce chapitre a pour but d'expliquer la méthodologie adoptée pour mener à bien le projet de recherche.

3.1 Objectif et démarche de recherche

En regard de la revue de littérature présentée plus tôt, l'objectif général de cette recherche est de **proposer un modèle de planification de projet agrégée avec chevauchement**.

Pour y arriver, notre démarche est la suivante : nous proposerons tout d'abord un modèle exact de planification agrégée avec chevauchement. Pour le tester, nous préparerons ensuite un ensemble d'instances de projets fictifs sur lesquelles nous testerons le modèle. Finalement, nous proposerons une heuristique plus adaptée que le modèle exact à des instances de grandes tailles.

Étant donné qu'un tel modèle n'a encore jamais été proposé, la méthode présentée ne sera pas destinée à être directement applicable en pratique. Il s'agira d'un premier modèle nécessitant encore des perfectionnements avant de pouvoir être utilisé dans de vrais projets. Pour cette même raison, les instances de tests utilisées seront fictives et l'analyse des résultats se concentrera plus sur les performances du modèle que sur les décisions de chevauchement.

Chacune des étapes de notre démarche sera expliquée plus en détails dans les sections suivantes.

3.2 Développement du modèle

Nous développerons tout d'abord un modèle qui s'appuie sur celui de Haït and Baydoun (2012) pour prendre en charge le chevauchement des lots de travaux. Le modèle de base a été initialement développé pour le problème RCPSVP. Il est ainsi bien adapté à la planification agrégée puisqu'il autorise les lots de travaux à avoir des intensités variables sur les périodes. Nous coderons ensuite le modèle avec le langage Optimization Programming Language (OPL).

3.3 Génération des instances de tests

Pour pouvoir tester notre modèle, nous devrons disposer d'un ensemble d'instances de projets. Étant donné qu'aucun ensemble d'instances de planification agrégée avec chevauchement existe, nous devrons générer nos propres instances. Pour ce faire, nous utiliserons des instances existantes de planification agrégée, sur lesquelles le modèle de base a déjà été testé. Puisque ces instances existantes ne contiennent pas de données relatives au chevauchement, il sera nécessaire de les modifier pour y inclure toutes les données manquantes. Ainsi nous serons en mesure de résoudre le modèle sur un grand nombre d'instances.

3.4 Expérimentation

Dans le but de valider le modèle, d'étudier sa performance et d'analyser les résultats obtenus, le modèle sera testé sur l'ensemble des instances avec différentes fonctions objectifs.

Nous développerons un code qui permet d'afficher les solutions obtenues sous forme de diagrammes et de graphes afin de les valider visuellement, ainsi qu'un code informatique pour une validation plus précise de la cohérence des valeurs obtenues pour chaque instance.

Ensuite, plusieurs coupes et alternatives de modélisation seront testées et leur impact sur les performances du modèle étudié pour retenir la combinaison la plus performante afin de perfectionner le modèle proposé.

Finalement, une étude statistique sur les performances et les résultats du modèle avec chevauchement sera conduite. D'abord, les impacts de paramètres des instances sur le temps de résolution et les performances du modèle seront étudiés et comparés entre eux. Ensuite, une analyse des résultats obtenus avec et sans chevauchement permettra de tirer des conclusions sur les avantages du chevauchement sur le coût et la durée des projets.

3.5 Développement d'une heuristique

Compte tenu de la complexité du problème RCCP avec chevauchement, il faut s'attendre à ce qu'aucune méthode exacte ne puisse résoudre des projets de grandes tailles dans des délais raisonnables. Nous essaierons alors de développer une heuristique basée sur notre modèle exact dans le but de fournir des solutions réalisables dans des temps de calcul acceptables.

3.6 Conclusion

Ce chapitre nous a permis d'expliciter la démarche que nous suivrons dans notre travail de recherche. Ce dernier prend son intérêt d'un besoin réel des entreprises de nouveaux outils de planification, mais aussi d'un manque détecté dans la littérature de satisfaire à leur besoin. Le chapitre suivant présentera le modèle proposé et les résultats obtenus dans un article scientifique soumis pour publication à *OR Spectrum*.

CHAPITRE 4 ARTICLE : A ROUGH-CUT CAPACITY PLANNING MODEL WITH OVERLAPPING

Georges Baydoun [1], Alain Haït [2], Robert Pellerin [3], Bernard Clément [4], and Guillaume Bouvignies [5]

Abstract

In this article, we propose an event-based mixed integer linear programming model for the Rough-Cut Capacity Planning (RCCP) problem with different feasible overlapping modes between work packages. In the model, time horizon is divided into time buckets used to evaluate resource usage, while starting and ending times for work packages are continuous. The model was tested on a benchmark of 5 sets of 450 instances each. More than half of tested instances were solved to optimality within 500 seconds. Results also prove that, while overlapping is more beneficial for accelerating project delivery times, it can still have a positive impact on project cost by allowing a better distribution of workload. Finally, overlapping options seem to have less influence on the performance of the model than project slack or number of work packages.

keywords : Rough-Cut Capacity Planning ; Concurrent Engineering ; Overlapping.

1. Department of Mathematics and Industrial Engineering, École Polytechnique, Montreal, Canada georges.baydoun@polymtl.ca

2. University of Toulouse, Institut Supérieur de l'Aéronautique et de l'Espace - Supaero, Toulouse, France alain.hait@isae.fr

3. Department of Mathematics and Industrial Engineering, École Polytechnique, Montreal, Canada robert.pellerin@polymtl.ca

4. Department of Mathematics and Industrial Engineering, École Polytechnique, Montreal, Canada bernard.clement@polymtl.ca

5. University of Toulouse, Institut Supérieur de l'Aéronautique et de l'Espace - Supaero, Toulouse, France guillaume.bouvignies@isae.fr

4.1 Introduction

Overlapping is a common practice used in construction and in product development projects to accelerate the execution of large projects. This technique consists in executing in parallel two sequential activities by allowing a downstream activity to start before the end of an upstream activity based on preliminary information. However, the overlapping of activities can entail rework tasks and modifications further to the transmission of complementary information after the start of the downstream activity (Berthaut et al., 2014). Activity overlapping thus allows reducing the total duration of project execution, at the expense of additional workload and execution cost associated to rework tasks.

Several authors have studied the relation between rework and the amount of overlap in project conducted in a concurrent engineering context or fast-tracking mode. For instance, Gerk and Qassim (2008) proposed a linear model for project acceleration using crashing, overlapping and activity substitution. Activity crashing includes the allocation of additional resources for an activity in order to accelerate its execution. Activity substitution is another technique for accelerating projects by replacing one (or more) activity by another activity.

Berthaut et al. (2014) and Grèze et al. (2012) proposed a more realistic approach by restricting overlapping possibilities to a set of feasible overlap durations for each couple of overlappable activities, instead of considering a continuous and linear relation between overlap amount and rework. This assumption is more realistic as overlapping points between activities are defined through clear document or information exchange in a concurrent engineering context, which limits the overlapping modes to a reduced and discrete set of possibilities.

However, these Resource-Constrained Project scheduling problem (RCPSP) models are not suited for planners in the early phases of projects as detailed activity content, durations, and resources are not known with precision and as work intensity cannot be assumed to be constant over execution time. Indeed, planners tend to adopt an aggregate planning approach in large engineering projects where Work Packages (WPs) are broadly defined as group of multiple activities that could extend on a long period, i.e. weeks or months (Cherkaoui et al., 2013). In practice, project planning is done by preparing several schedules at different phases of the project, where aggregation levels depend on the ongoing phase and on the targeted audience. For instance, the Front-End-Loading (FEL) approach, commonly used in large construction projects, is composed of successive planning stages. Early phases involve tactical planning based on Rough-Cut Capacity Planning (RCCP) techniques in order to fix all project milestones and estimate resource usage. At this level, projects are divided into work packages (WPs) which are clusters of activities. Rough-Cut Capacity Planning (RCCP)

models divide the planning horizon into time buckets (or periods) used to evaluate critical resource usage and by allowing resource allocation to WP to vary from one period to another De Boer (1998).

Recognizing the need of practitioners to better support the project planning function in the early phases of projects, this paper proposes an exact RCCP model that determines the order of execution in time of a set of WPs so as to minimize the total project duration and/or project cost, while respecting precedence relations, resource constraints and considering overlapping possibilities. The proposed model considers variable WP intensities and aggregate resource capacities.

The reminder of the paper is organized as follow. We first give a brief state of the art of existing RCCP models and overlapping models in section 4.2. We then introduce the original mixed-time RCCP model in section 4.3, before explaining our new RCCP model with multiple overlapping modes in section 4.4. Section 4.5 explains the generation of our test instances, and an illustrating example is presented in section 4.6. Finally, section 4.7 analyzes the performance and the results of our model, before giving some concluding remarks in section 4.8.

4.2 Related work

In the order acceptance stage of a project, companies tend to commit to due dates without an accurate knowledge of their resource capacities. The Rough-Cut Capacity Planning (RCCP) ensures, at an aggregated level, that the capacities of critical resources are sufficient to complete a project within its time and cost limits (De Boer, 1998). Performed at the tactical level, the RCCP is based on a horizon divided into time buckets (or periods) used to evaluate the resource usage. The WPs are defined by their work content and resource allocation can vary from one period to another. Capacity and resource allocation flexibility allow to adapt the WP durations according to time and cost-related considerations. WPs may start or end during a period hence it is possible to plan a WP and its successor within the same period.

Besides denominated RCCP models, RCPSP models where intensities can vary from period to period are also suitable for the RCCP problem as they consider fixed workload for each activity and variable resource usage between periods, and therefore variable activity durations. These models have several appelations in the literature such as Resource Constrained Project Scheduling with Variable Intensity Activities (RCPSVP) and RCPSP with flexible resource profiles.

Wullink (2005) distinguishes three classes of solution approaches for the RCCP : straight-

forward constructive heuristics, LP based heuristics and exact approaches. Among existing RCCP models, Hans (2001) proposed an exact approach that consists in determining the periods where each job can be executed, and then specifying the fractions of the WP contents that are actually executed in each period. However, this method can allow a predecessor and any direct successor to be performed in the same period without determining their ending and starting times within the period, which could lead to precedence infeasibility. Hans (2001) proposed two alternatives to avoid this problem by over-constraining the problem. Taking a project scheduling point of view, Kis (2005) proposed an RCPSP model with variable activity intensities, that forbids two activities with direct precedence relation to be executed in the same period.

More recently, Haït and Baydoun (2012) proposed an exact approach that consists of using continuous time representation for events, together with a discrete evaluation of resources. This means that start and ending dates of WPs can be determined within a time period, making it possible to guarantee the respect of precedence constraints. Yet, resource consumptions are still evaluated globally over periods, making the model suitable for planning at a tactical level where planning is performed in practice on a period-by-period basis (i.e. resource availabilities and allocations are determined per period).

Lately, Naber and Kolisch (2014) addressed the RCPSP with flexible resource profiles, meaning that resource usage of an activity can vary from period to period. They propose four different discrete-time MILP model formulations based on existing formulations and compare their performance. Their experiments show that the model called "FP-DT3", that is based on the RCPSP formulation of Bianco and Caramia (2013) and the RCPSVP model of Kis (2005), dominates the other formulations in both solution quality and run-time. Several heuristic approaches have also been proposed to solve the RCCP problem including constructive heuristics (De Boer, 1998) and linear-programming based heuristics (Gademann and Schutten, 2005).

Although some authors proposed interesting extensions to the precedence relations of RCPSVP models by allowing overlapping, none of these models considers overlapping and rework. In fact, Kis (2006) extended his RCPSVP model by introducing feeding precedence constraints : a successor can start after a percentage of the execution of his direct predecessor is completed. Alfieri et al. (2011) extended feeding precedence constraints to generalized precedence relations. However, both Kis (2006) and Alfieri et al. (2011) do not take reworks into account. Moreover, these models have only one possibility of precedence constraints and do not consider several feasible modes of overlapping with different amounts of rework.

Despite these recent advances, these models do not allow overlapping of WPs that are nor-

mally executed consecutively. This concurrent engineering method is a trend in product development and is also widely used in contexts such as construction in order to accelerate project execution. It also adds flexibility in starting and ending times, allowing a better usage of regular capacities thus reducing the need for external resources. However overlapping adds workloads, "reworks", on both down-stream and up-stream WPs (Berthaut et al., 2014). Reworks are due to information exchange and eventual alteration of the work, that are caused by starting a down-stream WP before all finalized information is available from up-stream WP.

To fill this gap, we propose a mixed integer linear-programming model, that is an extension of the RCCP model proposed by Haït and Baydoun (2012), where predecessor-successor WPs can overlap according to multiple overlapping modes. This extended model assumes that each overlapping mode can be defined by the percentage of execution of predecessor WP that needs to be reached in order to start the successor, as well as the amount of reworks on both WPs. The model is further explained in the following section.

4.3 Mixed-time RCCP model

This section presents the original model of Haït and Baydoun (2012). It combines a continuous time representation of events and a discrete time evaluation of resources. A set of binary variables ensures the relation between starting time, ending time and durations over periods. These durations give minimum and maximum workload that can be assigned to the period.

In this section only, the starting and ending times of WP i are denoted by t_i^0 and t_i^1, respectively. Binary variables z_{ip}^0 and z_{ip}^1 equal 1 if the starting and ending times (respectively) of WP i occur during or before time period p. Variables d_{ip} and l_{ip} are respectively the duration and the assigned workload of WP i over the period p. Finally, $Succ_i$ represents the set of successors of i. In section 4.4, the aforementioned notations are generalized in order to handle overlapping. Other definitions that are relevant to this section are given in tables 4.1 and 4.2.

In addition to basic definition-related constraints, constraints for the model of Haït and Baydoun (2012) are of four types : those ensuring the link between continuous and binary variables for starting and ending times, the ones concerning the durations over periods, scheduling constraints, and finally workload and intensity constraints.

4.3.1 Continuous/Binary variables constraints

The following constraints (4.1) to (4.3) ensure that binary variables z_{ip}^0 and z_{ip}^1 equal 1 only in the periods during or after starting and ending events (respectively). These binary variables

were first introduced by Pritsker and Watters (1968) as a step formulation for scheduling with limited resources, and used by several other authors including Bianco and Caramia (2013). However, only the model of Haït and Baydoun (2012) used them in the context of a mixed continuous and discrete time representation.

$$t_i^c \geq D \cdot p \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0,1\}, p \in P \tag{4.1}$$

$$t_i^c \leq D \cdot p + H \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0,1\}, p \in P \tag{4.2}$$

$$z_{ip+1}^c \geq z_{ip}^c \qquad \forall i \in I, c \in \{0,1\}, p \in \{1..|P|-1\} \tag{4.3}$$

4.3.2 Durations over periods

When considering a WP i and a time period p, starting and ending events can be before, during, or after p. Therefore, each couple WP i and time period p has six possible configurations, depicted in Figure 4.1.

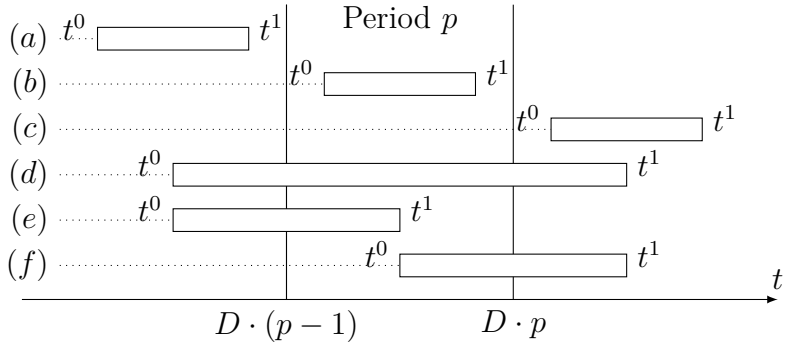

Figure 4.1 The six possible configurations for a WP regarding a time period p

The following constraints give the relations between WP durations over periods d_{ip}^1 on one side, and binary variables z_{ip}^0 and z_{ip}^1, and continuous time variables t_i^0 and t_i^1 on the other side :

$$d^1_{ip} \leq D \cdot (z^0_{ip} - z^1_{ip-1}) \qquad \forall i \in I, p \in 1..|P| \qquad (4.4)$$

$$d^1_{ip} \geq D \cdot (z^0_{ip-1} - z^1_{ip}) \qquad \forall i \in I, p \in 1..|P| \qquad (4.5)$$

$$d^1_{ip} \geq t^1_i - D \cdot p + D \cdot z^0_{ip-1} - H \cdot (1 - z^1_{ip}) \qquad \forall i \in I, p \in 1..|P| \qquad (4.6)$$

$$d^1_{ip} \geq D \cdot p \cdot (1 - z^0_{ip-1}) - t^0_i - D \cdot z^1_{ip} \qquad \forall i \in I, p \in 1..|P| \qquad (4.7)$$

$$\sum_{p=1}^{|P|} d^1_{ip} = t^1_i - t^0_i \qquad \forall i \in I \qquad (4.8)$$

Constraints (4.4) force d^1_{ip} to 0 when WP i is not active during period p (configurations (a) and (c)) and limit its value to period duration D if i and p are in any of the four other configurations. Inequalities (4.5), (4.6), and (4.7) concern configurations (d), (e), and (f) respectively. Inequalities (4.5) give d^1_{ip} lower-bounds if WP i begins before period p and is not completed during p. Constraints (4.6) (respectively(4.7)) provide lower-bounds for d^1_{ip} when WP i is finished (respectively started) in period p. Finally, constraints (4.8) provide global coherence between starting and ending times, and durations per periods.

4.3.3 Scheduling constraints

Each WP has its own time window, consisting of a release date RD_i and due date DD_i. Constraints (4.9) and (4.10) ensure that WP i is executed in its allowed time window. Moreover, a successor cannot start before its predecessor is completed (constraints (4.11)).

$$t^0_i \geq RD_i \qquad \forall i \in I \qquad (4.9)$$

$$t^1_i \leq DD_i \qquad \forall i \in I \qquad (4.10)$$

$$t^0_j \geq t^1_i \qquad \forall i \in I, j \in Succ_i \qquad (4.11)$$

4.3.4 Workload and intensity constraints

The workload assigned to a period should respect minimum and maximum allowed intensities BJ^{min}_i and BJ^{max}_i (constraints (4.12) and (4.13)), and the total assigned workload over time horizon should match the required workload for the WP (equalities (4.14)).

$$l^1_{ip} \geq BJ^{min}_i \cdot \frac{d^1_{ip}}{D} \qquad \forall i \in I, p \in P \qquad (4.12)$$

$$l^1_{ip} \leq BJ^{max}_i \cdot \frac{d^1_{ip}}{D} \qquad \forall i \in I, p \in P \qquad (4.13)$$

$$\sum_{p \in P} l^1_{ip} = L^{total}_i \qquad \forall i \in I \qquad (4.14)$$

4.3.5 Definition-related constraints

The following constraints are straight-forward results of definitions of *makespan*, regular and non regular resource allocations y^{int}_{rp} and y^{ext}_{rp}, and *cost*.

$$makespan \geq t^1_i \qquad \forall i \in I \qquad (4.15)$$

$$y^{int}_{rp} + y^{ext}_{rp} \geq \sum_{i \in I} A_{ir} \cdot l^1_{ip} \qquad \forall r \in R, p \in P \qquad (4.16)$$

$$y^{int}_{rp} \leq K^{int}_{rp} \qquad \forall r \in R, p \in P \qquad (4.17)$$

$$cost = C_{ext} \cdot \sum_{r \in R, p \in P} y^{ext}_{rp} \qquad (4.18)$$

Variables y^{int}_{rp} and y^{ext}_{rp} are respectively regular and non-regular allocations for resource r during period p. While we dispose of free K^{int}_{rp} regular capacity for resource r during period p (constraints (4.17)), non-regular capacities come at a price. Project cost is calculated as the total cost of non-regular capacities used for the whole project (equations (4.18)).

Possible objective functions are project makespan, project cost, or a trade-off between makespan and cost.

4.4 Mixed-time RCCP model with overlapping

Our new model is based on the same representation of time and events of WP start and WP end, but adds a third type of events : intermediate milestone attainment. These milestones are events that signal the attainment of a certain development state of a WP. They could correspond to important decision making moments, or the completion of key deliverables.

An overlapping mode between a WP i and its successor j is defined regarding the attainment – in terms of workload – of a milestone within WP i ; the successor can start once the milestone is reached. An overlapping mode for a WP i is a combination of different overlapping modes

between WP i and every successor of WP i that can overlap on i. Each WP i is therefore divided into $C_i + 1$ parts, where C_i is the number of successors of WP i that can overlap on it. The end of each part matches the time when a milestone of WP i is reached.

Take the example of a WP A with two successors that can overlap B and C. WP A is therefore devided into three parts ($C_A = 2$) in this example. WPs B and A have two overlapping modes : B can start after 70% or 100% (no overlapping) of A is completed in terms of workload. WPs C and A also have two overlapping modes at 35% and 100% of the total workload of A. This means that WP A has four overlapping modes with its successors as depicted in Figure 4.2. If overlapping mode 3 is selected for WP A, then variable e_{Am} equals 1 for $m = 3$ and 0 for $m \in \{1, 2, 4\}$.

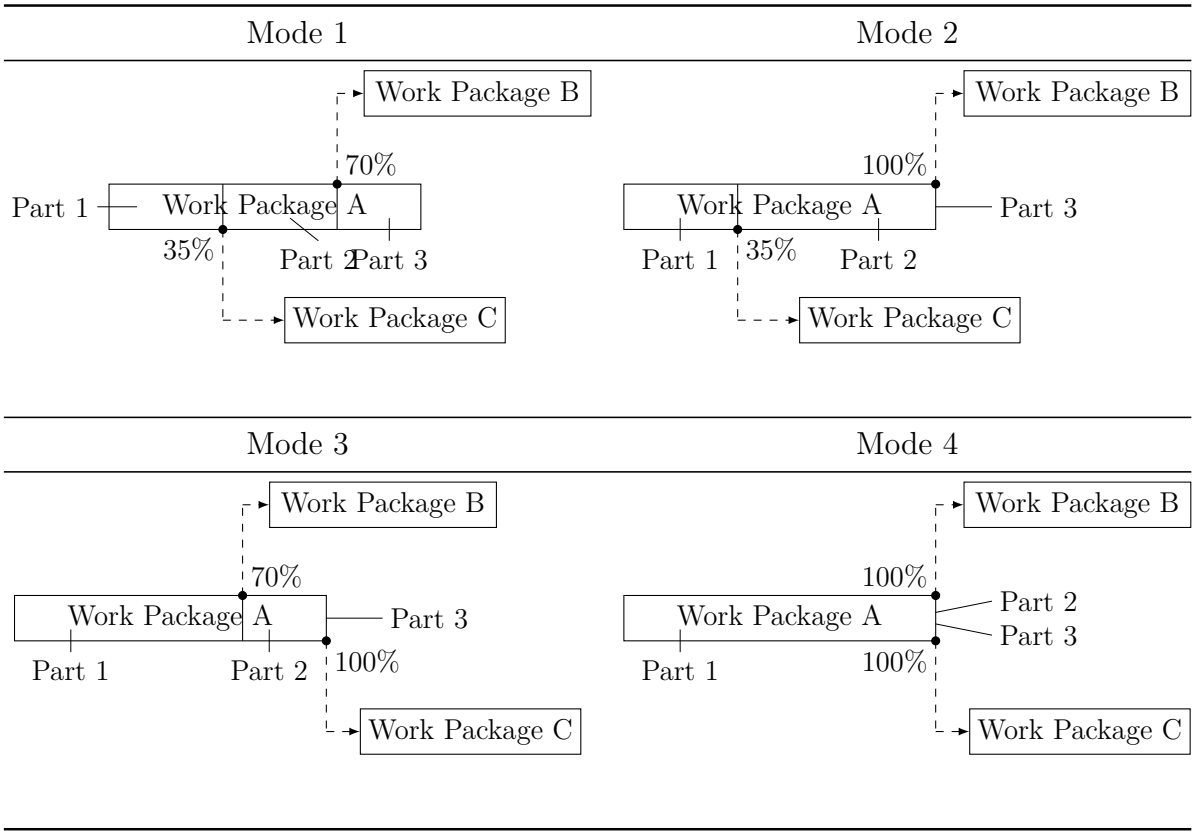

Figure 4.2 Four possible overlapping modes for a WP A with two successors B and C that can overlap on A

Each part c has its own set of durations over the periods, denoted d^c_{ip}, and assigned workload during the periods, l^c_{ip}. We also add overlapping durations over periods d_{ijp} for overlapping WPs. Variables d^c_{ip} and d_{ijp} guarantee that workloads (l^c_{ip}) and reworks ($l^{pred}_{ijp}, l^{succ}_{ijp}$) always respect the minimum and maximum allowed workload intensities. Tables 4.1 and 4.2 give the full nomenclature for our new RCCP model with overlapping.

Tableau 4.1 Nomenclature : sets & parameters

Set / Parameter	Description
P, D, H	Set of time periods ($p \in P$), duration of a period, time horizon $H = D \cdot \|P\|$
I	Set of work packages ($i \in I$)
RD_i, DD_i	Ready date (respectively due date) of i
R	Set of resources ($r \in R$)
K_{rp}^{int}	Available regular capacity of resource r during period p
A_{ir}	Percentage of consumption of resource r by WP i
BJ_i^{min}, BJ_i^{max}	Minimum and maximum workload that can be assigned for i during a duration of D
$Succ0_i$	Set of successors of i that cannot overlap on i ($j \in Succ0_i$)
$Succ1_i, C_i$	Set of successors of i that can overlap on i ($j \in Succ1_i$), $C_i = \|Succ1_i\|$
$Pred1_i$	Set of predecessors of i that can overlap on i ($i_0 \in Pred_i$)
M_i	Set of overlapping modes between i and its direct successors ($m \in M_i$)
Pos_{ijm}	Position of j among the successors of i according to mode $m \in M_i$
Ch_{ijm}	Binary parameter that equals 1 if WPs i and j overlap in mode $m \in M_i$
L_i^{total}	Total required workload for WP i
L_{im}^c	Required workload of part c of WP i in mode m
$L_{ijm}^{pred}, L_{ijm}^{succ}$	Required rework on i (respectively on j) in mode $m \in M_i$ due to overlapping
	between i and j
C_{ext}	Unitary cost of non-regular capacity

Tableau 4.2 Nomenclature : variables

Variable	Description
$t_i^0, t_i^{C_i+1}$	Starting time and ending time of i
t_i^c	For $c \in 1..C_i$: ending time of part c of i
z_{ip}^c	For $c \in 0..C_i + 1$: binary variable that equals 1 if t_i^c is in period p or before
e_{im}	Binary variable that equals 1 if mode m is chosen for i
d_{ip}^c	Duration of part c of i within period p
d_{ijp}	Duration of overlapping between i and j within period p
l_{ip}^c	Workload of part c of i during period p
$l_{ijp}^{pred}, l_{ijp}^{succ}$	Rework on predecessor i (resp. successor j) during p due to overlapping between i and j
$y_{rp}^{int}, y_{rp}^{ext}$	Regular and non-regular required capacity of resource of resource r in period p
$makespan$	Project makespan
$cost$	Project cost

4.4.1 Continuous/Binary variables constraints

Constraints (4.19), (4.20), and (4.21) are straight-forward results of the definition of z_{ip}^c, and generalize constraints (4.1) to (4.3).

$$t_i^c \geq D \cdot p \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0..C_i + 1\}, p \in P \tag{4.19}$$

$$t_i^c \leq D \cdot p + H \cdot (1 - z_{ip}^c) \qquad \forall i \in I, c \in \{0..C_i + 1\}, p \in P \tag{4.20}$$

$$z_{ip+1}^c \geq z_{ip}^c \qquad \forall i \in I, c \in \{0..C_i + 1\}, p \in \{1..|P| - 1\} \tag{4.21}$$

4.4.2 Durations over periods

Constraints (4.22) to (4.26) provide the link between durations per periods d_{ip}^c and variables t_i^c and z_{ip}^c, similarly to equations (4.4) to (4.8) of the original mixed-time model.

$$d_{ip}^{c} \leq D \cdot (z_{ip}^{c-1} - z_{ip-1}^{c}) \qquad \forall i \in I, c \in \ 1..C_i + 1, p \in 1..|P| \qquad (4.22)$$

$$d_{ip}^{c} \geq D \cdot (z_{ip-1}^{c-1} - z_{ip}^{c}) \qquad \forall i \in I, c \in \ 1..C_i + 1, p \in 1..|P| \qquad (4.23)$$

$$d_{ip}^{c} \geq t_i^c - D \cdot p + D \cdot z_{ip-1}^{c-1} - H \cdot (1 - z_{ip}^{c}) \qquad \forall i \in I, c \in \ 1..C_i + 1, p \in 1..|P| \qquad (4.24)$$

$$d_{ip}^{c} \geq D \cdot p \cdot (1 - z_{ip-1}^{c-1}) - t_i^{c-1} - D \cdot z_{ip}^{c} \qquad \forall i \in I, c \in \ 1..C_i + 1, p \in 1..|P| \qquad (4.25)$$

$$\sum_{p=1}^{|P|} d_{ip}^{c} = t_i^c - t_i^{c-1} \qquad \forall i \in I, c \in \{1..C_i + 1\} \qquad (4.26)$$

Constraints (4.27) to (4.29) concern overlapping durations within periods by giving upper-bounds for d_{ijp}. Constraints (4.27) and (4.28) make sure that overlapping durations within periods never exceed durations within periods of both predecessor i and successor j. Constraints (4.29) ensure that overlapping durations do not surpass the span between the starting time of j and ending time of i. Note that inequalities (4.27) consider the assumption that for a given WP, its overlapping predecessors can never overlap on its overlapping successors (no cascade effect).

$$d_{ijp} \leq d_{jp}^{1} \qquad \forall i \in I, j \in Succ1_i, p \in P \qquad (4.27)$$

$$d_{ijp} \leq \sum_{c=1}^{C_i+1} d_{ip}^{c} \qquad \forall i \in I, j \in Succ1_i, p \in P \qquad (4.28)$$

$$d_{ijp} \leq t_i^{C_i+1} - t_j^0 + H \cdot (1 - e_{im}) \qquad \forall i \in I, j \in Succ1_i, m \in \{M_i | ch_{im}^{j} = 1\}, p \in P \qquad (4.29)$$

4.4.3 Scheduling constraints

Constraints (4.30) and (4.32) force WP i to be in its allowed time window, while constraints (4.31) make sure that the different parts of WP i are in order. Moreover, the end of each part gives the time t_i^c when a successor in $Succ1_i$ can start. Constraints (4.33) ensure that successor j begins after the correct milestone of i. For successors in $Succ0_i$ that cannot overlap on i, a classical end-to-start constraint is defined in constraints (4.34). For successors in $Succ1_i$ that can overlap on i, constraints (4.35) exclude the case where a predecessor of i and a successor of i overlap (no cascade effect).

$$t_i^0 \geq RD_i \qquad \forall i \in I \tag{4.30}$$

$$t_i^{c+1} \geq t_i^c \qquad \forall i \in I, c \in \{0..C_i\} \tag{4.31}$$

$$t_i^{C_i+1} \leq DD_i \qquad \forall i \in I \tag{4.32}$$

$$t_j^0 \geq t_i^c - H \cdot (1 - e_{im}) \qquad \forall i \in I, m \in M_i, j \in Succ1_i, c \in \{1..C_i + 1 | c = Pos_{ijm}\} \tag{4.33}$$

$$t_j^0 \geq t_i^{C_i+1} \qquad \forall i \in I, j \in Succ0_i \tag{4.34}$$

$$t_j^1 \geq t_i^{C_i+1} \qquad \forall i \in I, j \in Succ1_i \tag{4.35}$$

Figure 4.3 presents an example of a WP (A) with two successors (B, C) that can overlap. A is therefore divided into three parts. In the depicted mode, the milestone corresponding to C comes before the one corresponding to B. Consequently C can start after the end of the first part of A, while B can start after the end of the second part of A.

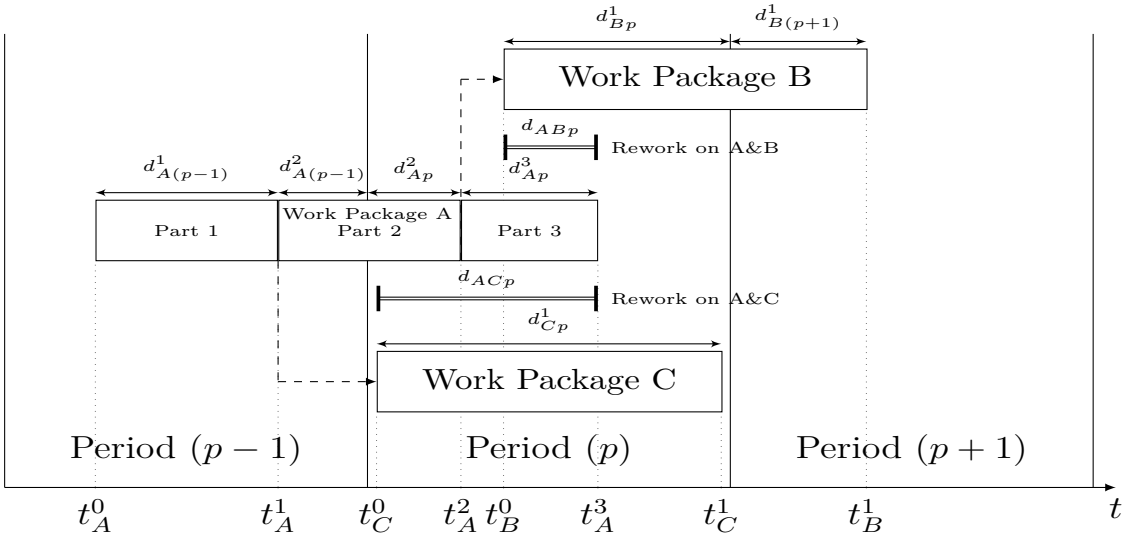

Figure 4.3 Work package (A) with two successors (B, C) that can overlap

4.4.4 Workload and intensity constraints

Constraints (4.36) and (4.37) ensure that the required workload is attained for each part of WP i according to the selected mode.

$$\sum_{p \in P} l_{ip}^{c} \geq L_{im}^{c} \cdot e_{im} \qquad \forall i \in I, c \in \{1..C_i + 1\}, m \in M_i \tag{4.36}$$

$$\sum_{p \in P, c \in \{1..C_i+1\}} l_{ip}^{c} = L_{i}^{total} \qquad \forall i \in I \tag{4.37}$$

In addition to initial workloads, reworks are required on predecessors and successors that overlap. The reworks should be executed during the overlapping time between predecessors and successors. Constraints (4.38) and (4.39) make sure that the total executed reworks match the required reworks in selected modes.

$$\sum_{p \in P} l_{ijp}^{pred} \geq L_{ijm}^{pred} \cdot e_{im} \qquad \forall i \in I, m \in M_i, j \in Succ1_i \tag{4.38}$$

$$\sum_{p \in P} l_{ijp}^{succ} \geq L_{ijm}^{succ} \cdot e_{im} \qquad \forall i \in I, m \in M_i, j \in Succ1_i \tag{4.39}$$

Workload variables $l_{ip}^{c}, l_{ijp}^{pred}$, and l_{ijp}^{succ} are linked to durations d_{ijp} and d_{ip}^{c} to ensure the respect of minimum and maximum allowed intensities. Constraints (4.40) and (4.41) guarantee that total assigned workload for WP i in period p stays in the allowed workload window for the total duration d_{ip}^{c} in period p.

$$\sum_{c=1}^{C_i+1} l_{ip}^{c} + \sum_{j \in Succ1_i} l_{ijp}^{pred} + \sum_{i_0 \in Pred1_i} l_{i_0 ip}^{succ} \leq BJ_i^{max} \cdot \frac{\sum_{c=1}^{C_i+1} d_{ip}^{c}}{D} \qquad \forall i \in I, p \in P \tag{4.40}$$

$$\sum_{c=1}^{C_i+1} l_{ip}^{c} + \sum_{j \in Succ1_i} l_{ijp}^{pred} + \sum_{i_0 \in Pred1_i} l_{i_0 ip}^{succ} \geq BJ_i^{min} \cdot \frac{\sum_{c=1}^{C_i+1} d_{ip}^{c}}{D} \qquad \forall i \in I, p \in P \tag{4.41}$$

Constraints (4.42) and (4.43) make sure that maximum and minimum intensities are respected for initial workload for every part of a WP, while constraints (4.44) and (4.45) ensure that maximum intensity is respected for reworks.

$$l_{ip}^{c} \leq BJ_{i}^{max} \cdot \frac{d_{ip}^{c}}{D} \qquad \forall i \in I, c \in \{1..C_i + 1\}, p \in P \qquad (4.42)$$

$$l_{ip}^{c} \geq BJ_{i}^{min} \cdot \frac{d_{ip}^{c}}{D} \qquad \forall i \in I, c \in \{1..C_i + 1\}, p \in P \qquad (4.43)$$

$$l_{ijp}^{pred} \leq BJ_{i}^{max} \cdot \frac{d_{ijp}}{D} \qquad \forall i \in I, j \in Succ1_i, p \in P \qquad (4.44)$$

$$l_{ijp}^{succ} \leq BJ_{j}^{max} \cdot \frac{d_{ijp}}{D} \qquad \forall i \in I, j \in Succ1_i, p \in P \qquad (4.45)$$

4.4.5 Definition-related constraints

The following constraints are straight-forward results of the definition of variables e_{im}, y_{rp}^{int}, y_{rp}^{ext}, *makespan*, and *cost*.

$$\sum_{m \in M_i} e_{im} = 1 \qquad \forall i \in I \qquad (4.46)$$

$$makespan \geq t_{i}^{C_i+1} \qquad \forall i \in I \qquad (4.47)$$

$$y_{rp}^{int} + y_{rp}^{ext} \geq \sum_{i \in I} A_{ir} \cdot \Big(\sum_{c=1}^{C_i+1} l_{ip}^{c} + \sum_{j \in Succ1_i} l_{ijp}^{pred} + \sum_{i_0 \in Pred1_i} l_{i_0ip}^{succ} \Big) \qquad \forall r \in R, p \in P \qquad (4.48)$$

$$y_{rp}^{int} \leq K_{rp}^{int} \qquad \forall r \in R, p \in P \qquad (4.49)$$

$$cost = C_{ext} \cdot \sum_{r \in R, p \in P} y_{rp}^{ext} \qquad (4.50)$$

Possible objective functions are project makespan, project cost, or a trade-off between makespan and cost.

4.5 Instances generation

4.5.1 Original test instances

In order to test our model, we used a set of 450 instances that were generated by De Boer (1998) and that are commonly used to test RCCP models. Each instance consists in one project and was generated randomly applying a procedure developed by Kolisch et al. (1995).

Using three parameters, De Boer (1998) generated 45 classes of 10 instances each. A class is characterized by the number of WP n, the total number of resources r, and its average slack s. The latter parameter is defined as follow : $s = \frac{\sum_{j \in I} DD_j - RD_j - Dmin_j + 1}{n}$. Three different

values are possible for n (10, 20, or 50 activites), and for r (3, 10, or 20 resources), while parameter s has five possible values (2, 5, 10, 15 and 20).

The following section explains how we modified the instances of De Boer (1998) so as to allow overlapping.

4.5.2 Modified test instances

We created a program that modifies the original instances of De Boer (1998) in order to handle overlapping. In addition to the previous parameters n, r, and s, we introduced six more parameters for our modified instances generation. These parameters are defined in Table 4.3.

If $p_{\%} = 0$, $Succ0_i$ contains all the successors of WP i and $Succ1_i$ is empty for every WP i. This will result in having instances that are equivalent to the original instances. On the other hand, if $p_{\%} = 1$, $Succ1_i$ contains all the successors of WP i and $Succ0_i$ is empty for every WP i.

If $0 < p_{\%} < 1$, then we randomly choose $\lceil p_{\%} \cdot Total\,Number\,Of\,Couples \rceil$ of the couples to overlap. If the successor is in $Succ1_i$, the couple (i, j) will then have a random number of overlapping modes between Nb_{min} and Nb_{max}.

If a couple $(i, j)|j \in Succ1_i$ has Nb overlapping modes, then the possible overlapping modes would be to begin the successor j after :

$$(100 - n \cdot Ov) \cdot L_i^{total} \qquad \forall n \in 1..Nb \tag{4.51}$$

Finally, for each mode, the needed rework is calculated as follow :

$$NeededRwk_i = (n \cdot Ov) \cdot \alpha_{rwk}^{pred} \cdot L_i^{total} \qquad \forall n \in 1..Nb \tag{4.52}$$

$$NeededRwk_j = (n \cdot Ov) \cdot \alpha_{rwk}^{succ} \cdot L_j^{total} \qquad \forall n \in 1..Nb \tag{4.53}$$

For the reminder of this papaer, we fixed $Nb_{min} = 2$, $Nb_{max} = 3$, $Ov = 20$, and $\alpha_{rwk}^{pred} = \alpha_{rwk}^{succ} = 0.4$ for all generated instances, and only varied parameter $p_{\%}$.

In addition to adding data related to overlapping, we recalculated the release and due dates of the WPs. In fact, time windows were tightened in the original instances, considering simple Finish-to-Start precedence relations De Boer (1998). This makes them inadequate for our instances that allow overlapping. We explain in details our calculations of the release and

Tableau 4.3 Six new parameters for the modified instances generation

Parameter	Description
$p_{\%}$	Percentage of predecessor-successor couples that can overlap. $0 \leq p_{\%} \leq 1$.
Nb_{min}	Minimum number of overlapping modes for every predecessor-successor couple that can overlap. $Nb_{min} \geq 1$.
Nb_{max}	Maximum number of overlapping modes for every predecessor-successor couple that can overlap. $Nb_{max} \leq \frac{100}{Ov}$.
Ov	Percentage that characterizes the amount of overlapping. $0 < Ov \leq 100$.
α_{rwk}^{pred}	Coefficient that characterizes the quantity of needed rework on predecessors. $0 \leq \alpha_{rwk}^{pred} \leq 1$.
α_{rwk}^{succ}	Coefficient that characterizes the quantity of needed rework on successors. $0 \leq \alpha_{rwk}^{succ} \leq 1$.

due dates for our modified instances in section 4.5.2.

Calculations of release and due dates

In the instances of De Boer (1998) time windows were first calculated using longest path calculations. They were then tightened in order to respect a maximum slack and an average slack. After each modification, all release and due dates were updated using longest path calculations.

For our modified instances, we first spotted all ready and due dates that were not obtained via longest path calculations in the original instances, and we fixed them in our modified instances. We then tightened all time windows using longest path calculations adapted to overlapping. These calculations give for a WP the earliest start date (respectively latest finish date) knowing the earliest start dates (respectively latest finish dates) of all its predecessors (respectively successors) and the maximum possible overlapping with each predecessor (respectively successor). With our adapted calculations of release and due dates, we obtain the same time windows as the original instances when fixing $p_{\%} = 0$ (no overlapping), and possibly larger time windows when $p_{\%} > 0$.

Note that modifying time windows increases the slack values of the instances of De Boer (1998). We calculated the new slack values for the highest $p_{\%}$ value of our new instances (worst case scenario), and found out that the difference between the new and the original values, is equal to 2.35 on average, with a 99% confidence interval of $[2.18; 2.52]$. This means that even in the worst case, our new instances can still be grouped according to the slack value of their original instances. For the reminder of this paper, and for clarity's sake, we will

keep using the original slack values as parameters for our modified instances.

4.6 Illustrating example

In this section, only two simple instances (derived from rccp192) are presented for illustration purpose. For each instance, the project consists of 10 WPs, 3 resources, and a time horizon of 23 weeks. The first instance was generated while fixing $p_{\%} = 0$, and the second one was created with $p_{\%} = 0.2$. Figure 4.4 shows the precedence relations between WPs for the two instances. Each vertice represents a WP, and each directed edge represents a predecessor-successor relationship between two WPs. When a predecessor-successor couple has two or more overlapping modes, a label appears on the edge showing the number of overlapping modes between the two WPs.

In the case where $p_{\%} = 0$, no overlapping is allowed between any couple. Thus, all the 12 predecessor-successor couples have one overlapping mode (no overlapping). When $p_{\%} = 0.2$, the instance generator chooses $\lceil 0.2 \cdot 12 \rceil = 3$ couples and randomly gives 2 or 3 overlapping modes for each chosen couple. As it appears in Figure 4.4, couples WPs 1-4 and WPs 6-10 have 2 overlapping modes each : WPs 4 and 10 can both start after 80%, or 100% of WPs 1 and 6 (respectively) are completed. Couple WPs 2-5 have 3 overlapping modes : WP 5 can start after 60%, 80%, or 100% of WP 2 is attained. Thus WPs 1 and 6 have 2 overlapping modes with their successors, while WP 2 has 3 overlapping modes. Furthermore, each of WPs 1, 2 and 6 is divided into 2 parts, as they only have one successor that can overlap.

We implemented our model using the Optimization Programming Language (OPL) and ran our model on IBM ILOG CPLEX Optimization Studio 12.5.1.0 while minimizing a tradeoff between project cost and makespan. We also implemented a code in order to visualize the solution with a Gantt chart.

Figure 4.5 shows the charts that we obtained with both instances, as well as the allocations for the three resources. WPs are shown in descending order in the Gantt chart (WP 1 is the highest, and WP 10 is the lowest). For each WP, a white rectangle delimits the allowed time window. Note that these rectangles are wider for $p_{\%} = 0.2$ because ready and due dates were recalculated in order to take into account the possibility of overlapping. Colored rectangles show starting and ending dates of WPs (or possibly of parts of WPs). Resource allocations are shown per period with stacked bar charts, where each color relate to a WP in the Gantt chart, and where hatched bars refer to rework. Stepped curves represent regular resource availability for each period.

Both instances were solved to optimality in less than 4 seconds. When allowing overlapping,

project makespan was reduced at the price of increasing project cost. In Figure 4.5, we see that WP 1 overlaps on WP 4 thus entailing rework workloads for both WPs. This additional workload is allocated during period 6. Note that, because of overlapping, WPs 8 and 9 finished earlier, thus reducing project duration by 2.3 periods (11%). Meanwhile project cost is increased by 43.3 units (34%) because of a greater use of non regular resources.

However, minimizing tradeoff function does not always have the same effect on project cost and makespan as in this example. Other instances showed that overlapping can reduce the use of non regular resources, thus diminishing project cost, at the expense of an increased project duration.

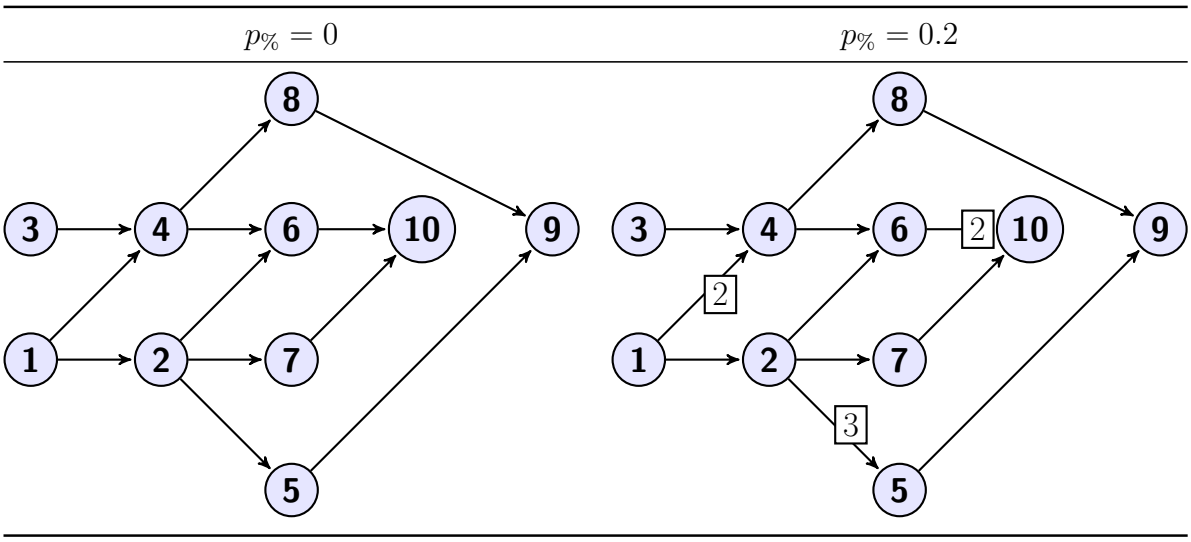

Figure 4.4 Precendence relations between WPs for a project with and without overlapping

4.7 Computational results

We performed all our tests on a single thread, using a computational grid consisting of 26 PCs with two 3.07 GHz Intel(R) Xeon(R) X5675 Processors under Linux. We encoded our model using the Optimization Programming Language (OPL) and ran our model on IBM ILOG CPLEX Optimization Studio 12.5.1.0 using the default values for all CPLEX parameters. We first ran our model on five sets, with five different values of $p_{\%}$ (0, 0.1, 0.2, 0.3 and 0.4), of 450 instances each (total of 2250 instances), with a time limit of 10000 seconds for every test. We then divided the time limit into 20 intervals of 500 seconds each, and counted the number of times CPU time was inside each interval. Figure 4.6 shows that 56.9% of instances were solved to optimality during the first 500 seconds, and that time limit was reached before optimality for 32.2% of instances. Figure 4.6 also shows that only 2% of instances

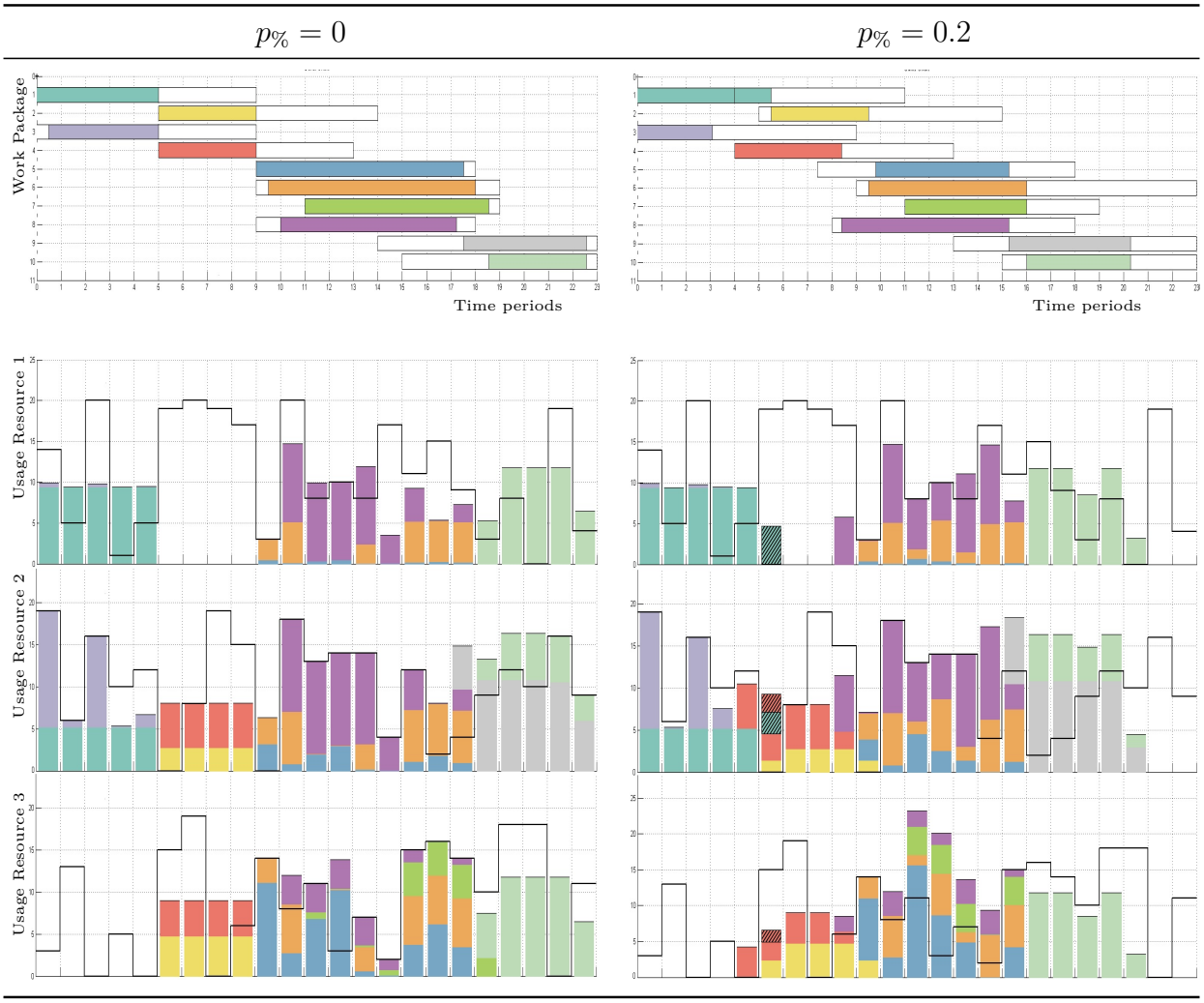

Figure 4.5 Gantt chart, with usage per period of the three resources

were solved to optimality between 5000 and 10000 seconds. Consequently, in the remaining tests, we terminated the search after 5000 seconds for every test.

4.7.1 Performance analysis

A series of tests was conducted in order to evaluate the performance of our new model. We denote by A the original model of Haït and Baydoun (2012) and B our new model that handles overlapping. We ran the model A on the 450 instances of De Boer (1998), and our model B on our modified instances of De Boer (1998) where we fixed $p_{\%} = 0$. In this particular case where no overlapping couples are allowed, the two sets of instances are equivalent. Having the same optimal solutions, we only compare the performance of the two models in terms of CPU times, the gaps between lower and upper bounds, and the number of times the models proved

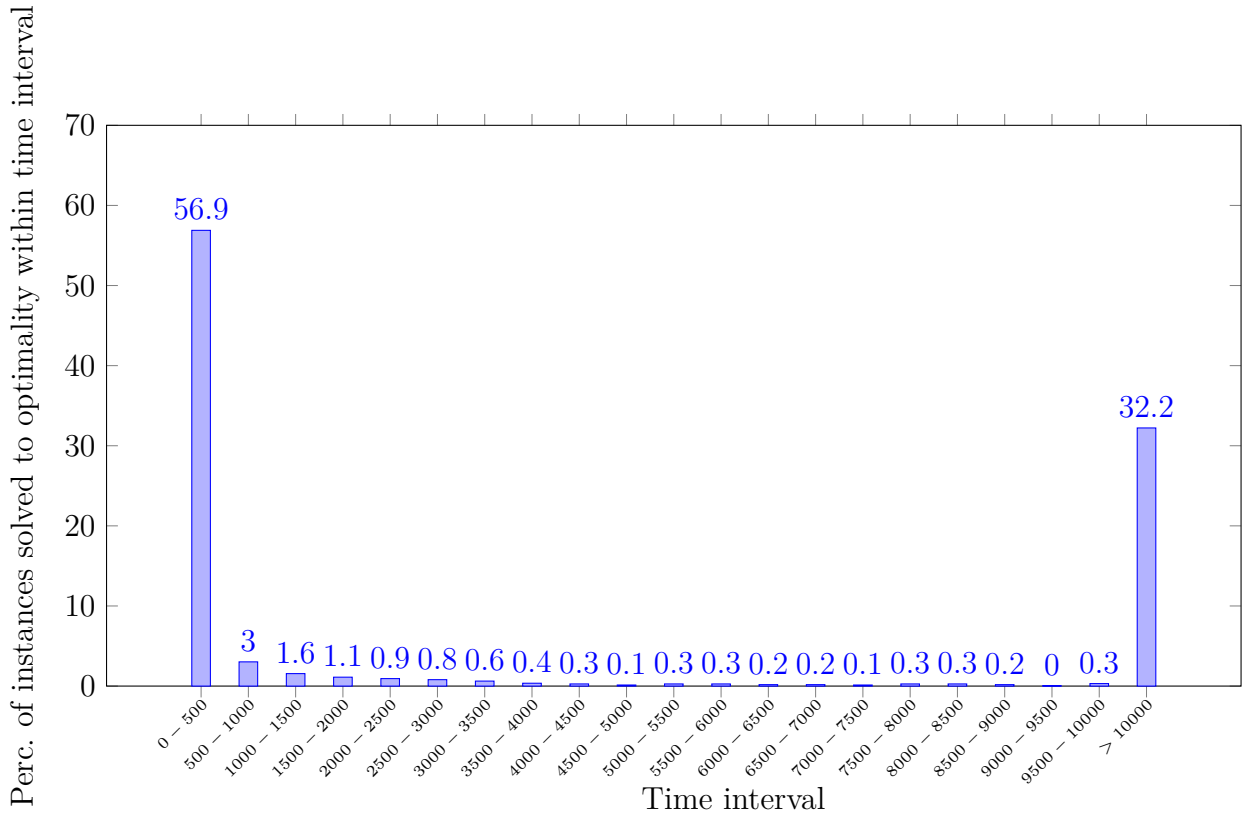

Figure 4.6 Percentage of instances that were solved to optimality within the specified time interval

optimality. Table 4.4 shows for each class the average CPU time for the original model A and our new model B, with the objective of minimizing project cost, while Table 4.5 compares the number of times both models A and B found an optimal solution for each class. Tables 4.4 and 4.5 show that the performance of new model B is not degraded compared to model A when tested on equivalent instances.

We also tested our new model B on five sets of instances with different values of $p_{\%}$, when minimizing project cost. We then defined five criteria for evaluating the performance of the model for each value of $p_{\%}$: the percentage of instances that were not solved to optimality, the average CPU time divided by time limit, the percentage of instances that needed more than 50% of the time limit in order to be solved to optimality, the average gap, and the percentage of instances that have more than 5% of gap when reaching time limit. Figure 4.7 presents the results for each value of $p_{\%}$ in a radar chart, where better performing set of instances are closer to the center. Figure 4.7 shows that the performance of our model is degraded on average, according to the five defined criteria, when the percentage of predecessor-successor

Tableau 4.4 Average CPU time in seconds for models A and B when minimizing the project cost without overlapping

	$n = 10$			$n = 20$			$n = 50$		
$r =$	3	10	20	3	10	20	3	10	20
$s = 2$									
A	0.1	0.1	0.1	0.1	0.2	0.2	0.4	0.6	1
B	0.1	0.1	0.1	0.2	0.2	0.2	0.7	0.9	1.1
$s = 5$									
A	0.3	0.4	0.4	1	2.1	3.8	14	289	280
B	0.3	0.4	0.5	1.6	2.5	4	30	406	346
$s = 10$									
A	3.5	2.5	3	39	223	235	2694	4930	5000
B	4.9	2.7	3.1	56	383	187	3488	5000	5000
$s = 15$									
A	6	10	22	450	2147	3115	5000	5000	5000
B	11	12	21	636	2217	2980	5000	5000	5000
$s = 20$									
A	13	66	48	1036	4198	3818	5000	5000	5000
B	22	74	48	2071	4315	3771	5000	5000	5000

Tableau 4.5 Number of times models A and B proved optimality when minimizing the project cost without overlapping

	$n = 10$			$n = 20$			$n = 50$		
$r =$	3	10	20	3	10	20	3	10	20
$s = 2$									
A	10	10	10	10	10	10	10	10	10
B	10	10	10	10	10	10	10	10	10
$s = 5$									
A	10	10	10	10	10	10	10	10	10
B	10	10	10	10	10	10	10	10	10
$s = 10$									
A	10	10	10	10	10	10	6	1	0
B	10	10	10	10	10	10	4	0	0
$s = 15$									
A	10	10	10	10	7	7	0	0	0
B	10	10	10	10	8	6	0	0	0
$s = 20$									
A	10	10	10	10	4	3	0	0	0
B	10	10	10	8	2	4	0	0	0

couples that can overlap augments. This is the result of an increase in the number of binary and continuous variables, due to the separation of some WPs into several parts, and also due to the different possible overlapping modes.

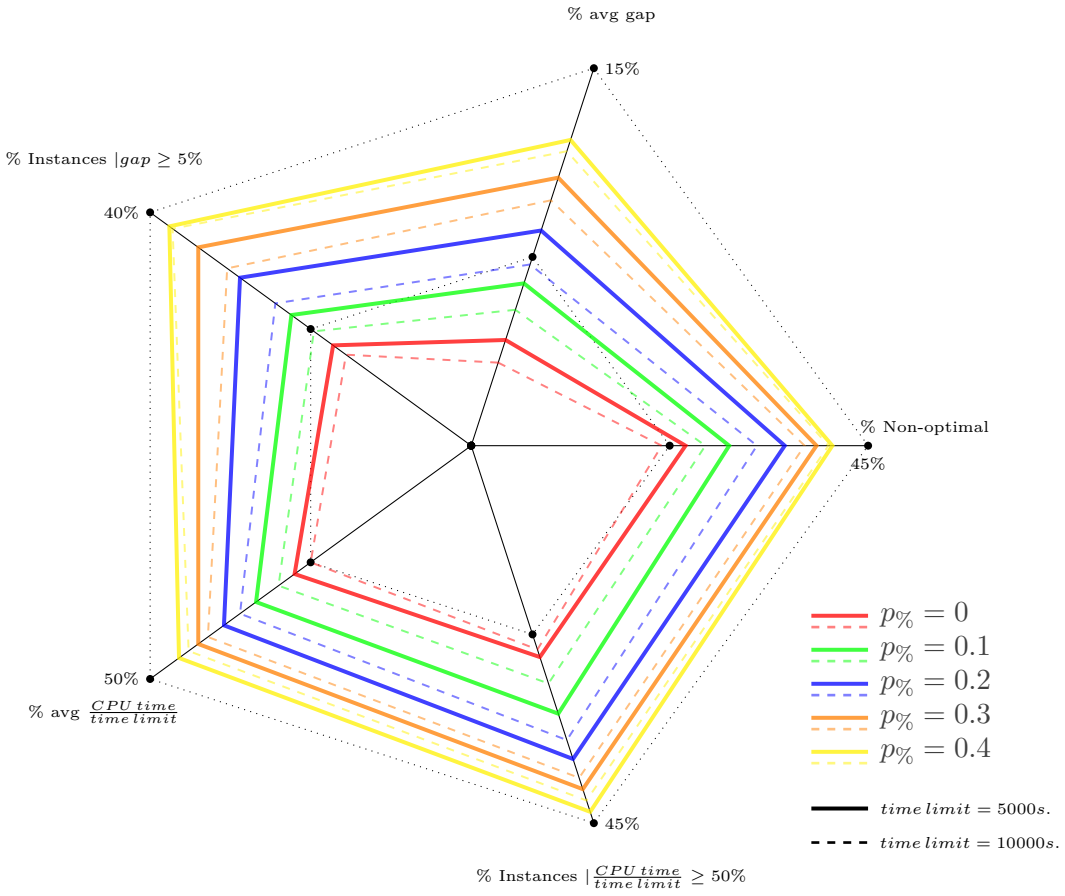

Figure 4.7 Impact of changing $p_{\%}$ on the performance of B, when minimizing project cost

It is clear that the new parameter $p_{\%}$ is not the only factor that has an impact on CPU time ; all the four aforementioned parameters of instances can affect the performance of the model. Figure 4.8 shows for each parameter s, r, n and $p_{\%}$, the average CPU time, with a 95% confidence interval. It suggests that parameters n and s have the biggest impact on the performance, followed by $p_{\%}$, and lastly r.

We first studied the effects of parameters s, r, n and $p_{\%}$, on having a CPU time bigger than time limit. For that matter, we created a binary variable that separates cases where CPU time is bigger than time limit (denoted event E_1), from those where CPU time is smaller (denoted event E_2). We then performed a Logistic regression on this binary variable, with parameters s, r, n and $p_{\%}$ as continuous predictors. The Logistic regression correctly predicts 94% of events E_2, and 82% of events E_1. The area under ROC curve is 0.964 (Figure 4.9),

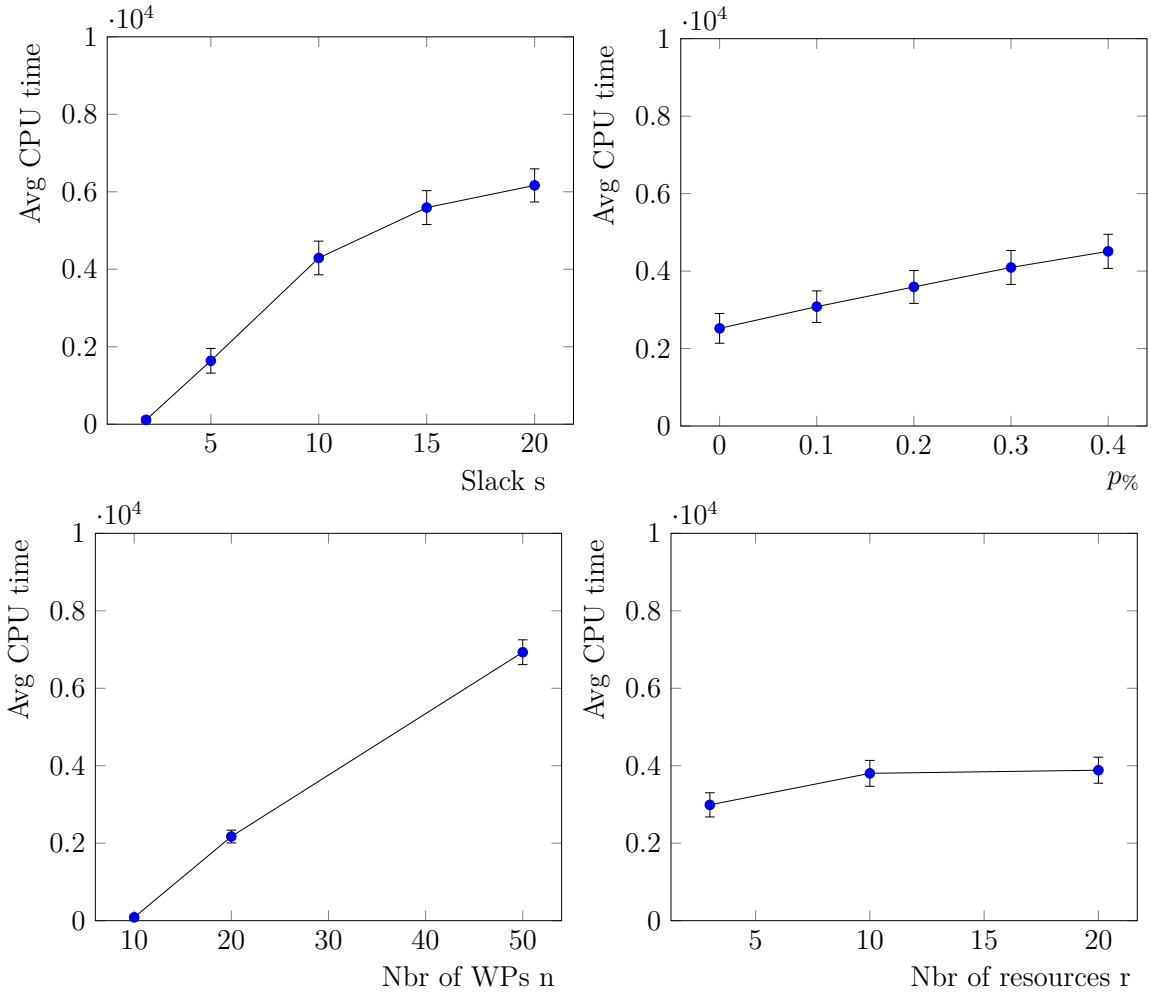

Figure 4.8 Average CPU time, with 95% confidence level for different values of four instance parameters

meaning that the regression has good accuracy. Odds ratios in Table 4.6 show the relative ratios of odds of event E_1 when increasing the predictor of one unit. All four values are bigger than 1, meaning that increasing any parameter augments the probability of event E_1. Also note that a unitary increase of s or n has significantly more impact on odds of event E_1 than a unitary increase of parameters r or $p_{\%}$.

We also conducted an ANOVA study on 1522 instances where CPU time did not reach time limit. This study gives results that are in line with our assumptions. In fact, Pareto chart of standardized effects (Figure 4.10) shows that factor s has the most significant impact on CPU time, followed by n, $p_{\%}$, and lastly r.

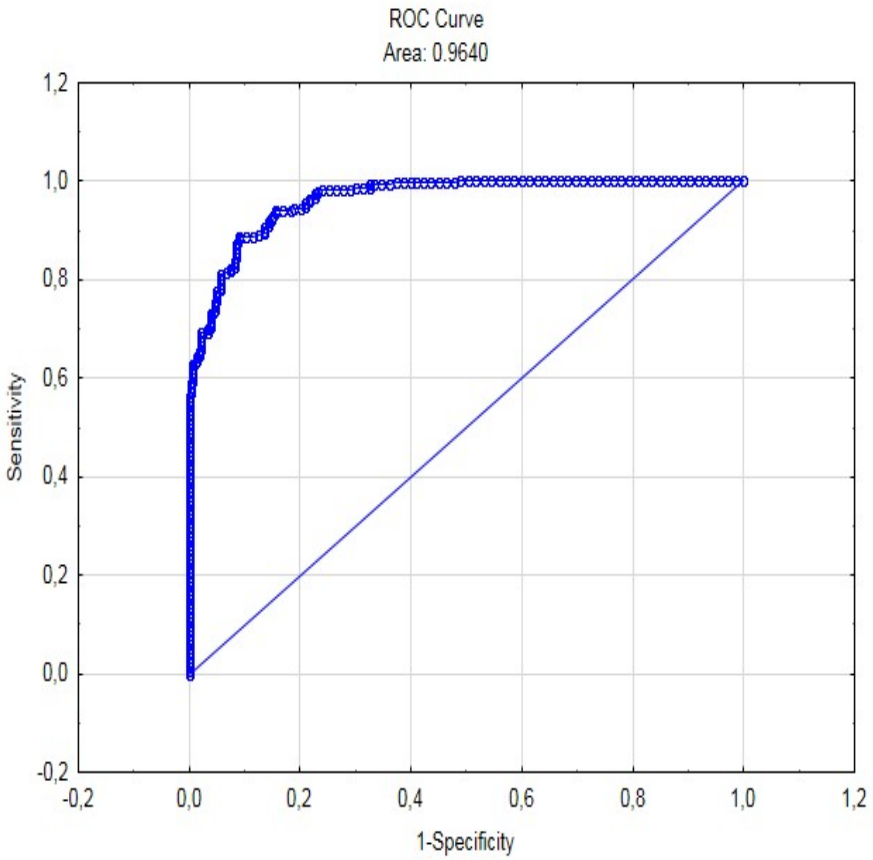

Figure 4.9 Receiver Operating Characteristic (ROC) curve for the Logistic regression

4.7.2 Results analysis

In order to conduct an analysis on the results that were obtained, we ran our model on the 450 instances with $p_{\%} = 0$ (equivalent to the original instances of De Boer (1998)) two times. We first fixed an objective of minimizing project cost, and then we ran the model another time while minimizing project delivery time. Among 450, only 340 were solved to optimality when minimizing project cost. For these instances, we obtained the optimal cost $Cost_0$ and makespan $Makespan_0$. We integrated these two parameters in all the corresponding data files (with the five different values of $p_{\%}$). This means that for a given instance, we added two additional parameters, $Cost_0$ and $Makespan_0$, which are the minimum possible cost and makespan for the instance if no overlapping was allowed.

We used these two parameters in order to create a tradeoff objective function (4.54) with normalized project cost and delivery time.

Tableau 4.6 Odds ratios of event E_1 for a unitary increase in parameters

Parameter	Odds ratio
s	1.645707
n	1.226389
r	1.079625
$p_{\%}$	1.075126

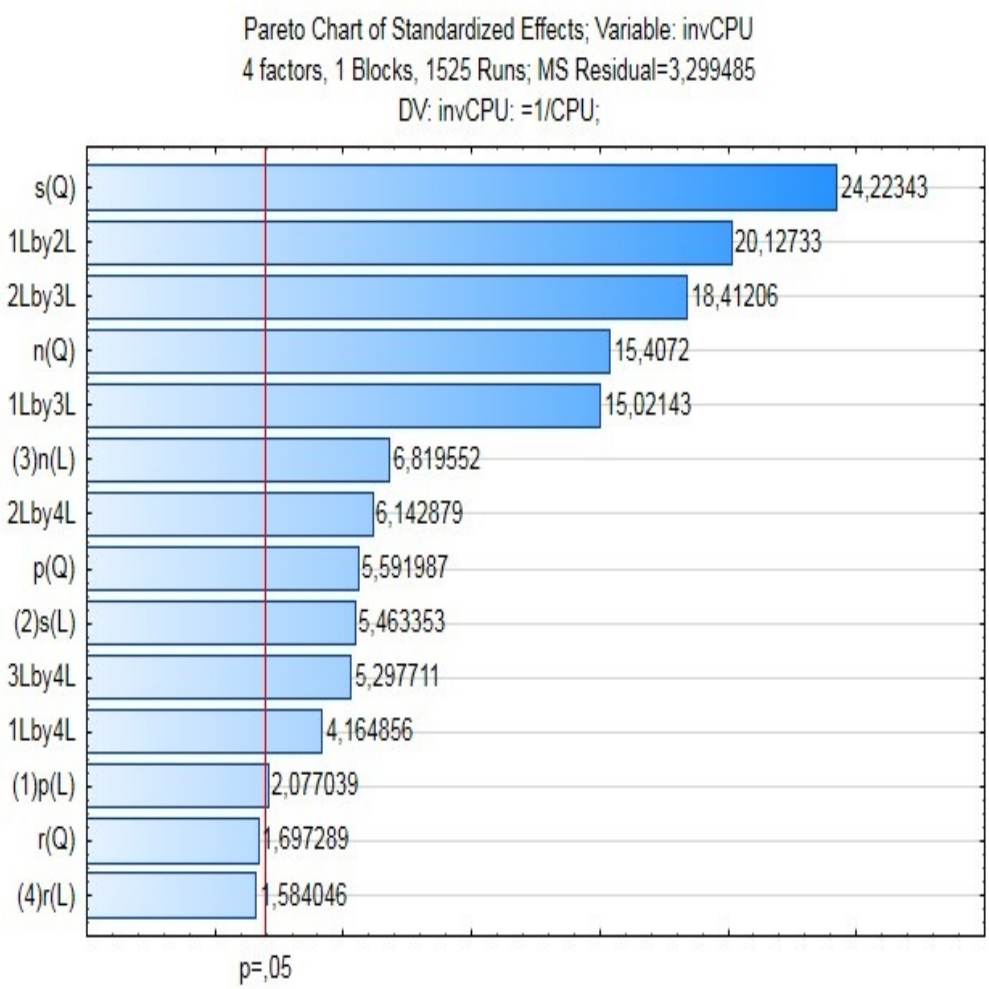

Figure 4.10 Pareto chart of standardized effects for factors n, s, $p_{\%}$, and r, and dependent variable $log(1/CPU\,time)$

$$minimize \quad \beta_{cost} \cdot \frac{cost}{Cost_0} + \beta_{makespan} \cdot \frac{makespan}{Makespan_0} \tag{4.54}$$

For each instance we tested three combinations for the coefficients β_{cost} and $\beta_{makespan}$ as

depicted in Table 4.7. For each combination, and each non-zero value of $p_{\%}$, we compared the results to the case where no overlapping was allowed ($p_{\%} = 0$). Figure 4.11 shows for each combination and non-zero value of $p_{\%}$, the percentage of instances where the objective was improved compared to the case where no overlapping was allowed.

Two remarks can be made concerning the shape of the bar graph 4.11. First, note that, for each combination of values for β_{cost} and $\beta_{makespan}$, the percentage of instances where the objective was improved when allowing overlapping, increases with $p_{\%}$. This can be explained by the fact that the bigger $p_{\%}$ is, the more flexible the problem becomes. Thus, it is more likely to have a smaller objective value with bigger $p_{\%}$. Secondly, note that for a given $p_{\%}$, the percentage of instances that have better objective with overlapping is bigger when $\beta_{cost} = 25$ and $\beta_{makespan} = 75$ (combination 1) than the case where $\beta_{cost} = 75$ and $\beta_{makespan} = 25$ (combination 3). This can be explained by the fact that overlapping is mainly beneficial when accelerating project delivery time. Thus its favorable effects are more pronounced when the tradeoff puts more emphasis on makespan than on cost. However, even though overlapping entails additional workload, it is still beneficial for having smaller project cost, by adding a flexibility in distributing the workload during regular resource capacities.

4.8 Conclusion

Motivated by the common use of concurrent engineering methods in construction projects, we proposed an interesting extension of the RCCP that allows overlapping of WPs. Five sets of 450 modified RCCP instances were created and let us conduct two types of analysis on our model. The performance analysis showed that our model is concurrent with the original model when our modified instances are equivalent to the original ones, and becomes less efficient when the percentage of possible overlapping couples is increased. However, this slower performance is the price to pay in order to have smaller project delivery time and cost, as it was shown by our results analysis. In fact, our results showed that overlapping is beneficial for accelerating project delivery times, and also for reducing project cost by allowing a better distribution of workload.

Tableau 4.7 Three combinations for coefficients β_{cost} and $\beta_{makespan}$

	β_{cost}	$\beta_{makespan}$
Combination 1	25	75
Combination 2	50	50
Combination 3	75	25

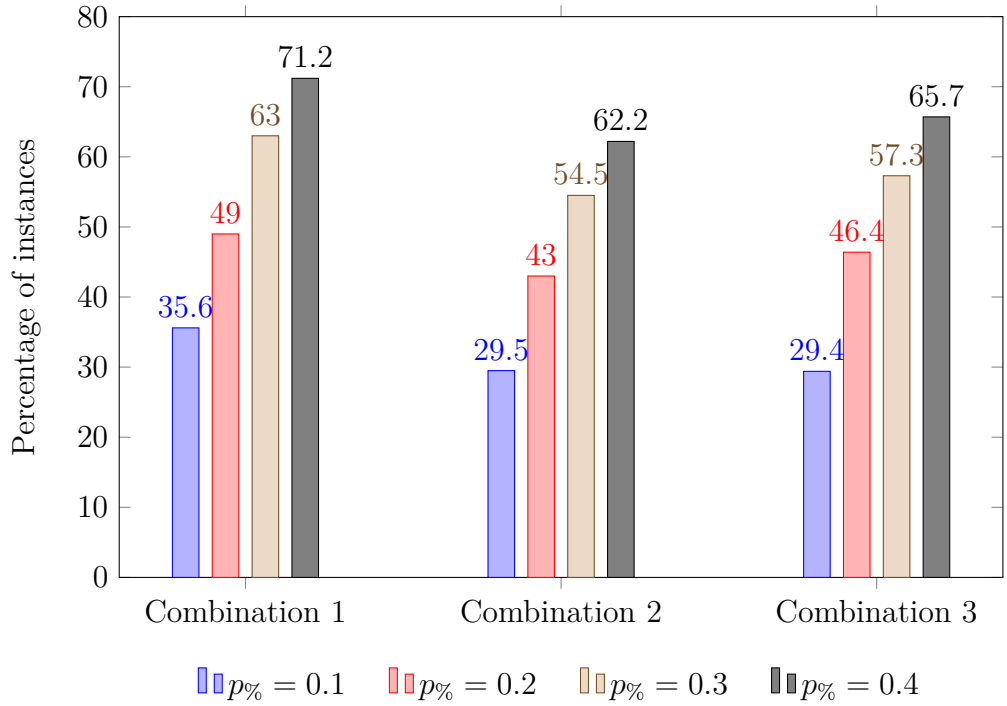

Figure 4.11 Percentage of instances where overlapping improved the objective

Future work could focus on ameliorating the performance of the model, by exploring alternative modeling and solving techniques. Another avenue for research could be the development of a model-based heuristic dedicated for the hard instances, in order to give acceptable solutions in reasonable time.

acknowledgements

This work has been supported by the Natural Sciences and Engineering Research Council of Canada and the Jarislowsky/SNC-Lavalin Research Chair in the Management of International Projects. Their support is gratefully acknowledged.

CHAPITRE 5 DISCUSSION

Ce chapitre examine les limitations de l'approche proposée, suggère des améliorations possibles, propose une solution alternative et discute des implications pratiques de la recherche dans un cas d'utilisation réelle.

5.1 Limitations de la solution proposée

Les limitations de la solution proposée concernent les hypothèses considérées, les instances de test utilisées et la résolution exacte par programme linéaire en nombres entiers :

- Nous considérons que les quantités de retouches sur les prédécesseurs et les successeurs qui chevauchent sont des données du problème. Or, les retouches représentent des efforts accrus d'échange et de communication entre les lots de travaux qui chevauchent et des risques d'erreurs et de retravail sur le successeur. Ainsi, les quantités de retouches peuvent être très aléatoires et difficiles à estimer par les gestionnaires au moment de la planification.
- Nous considérons dans le modèle un seul type de relation d'antériorité pour le chevauchement (type *%Completed-to-Start*). Or, il est possible en pratique d'envisager des relations d'antériorité généralisées avec chevauchement, semblables à celles proposées par Alfieri et al. (2011). Un des avantages de notre modèle est qu'il permet facilement l'intégration de telle relations.
- Toutes les données du modèle proposé sont considérées déterministes. Or les charges de travail des lots ne sont qu'une estimation pouvant subir différentes variations lors de l'exécution. De plus, les quantités de retouches incluent des risques d'erreurs et de retravail. Ainsi, une approche stochastique pourrait être plus réaliste pour notre problème.
- L'essai de notre modèle sur des instances fictives ramène d'autres limitations sur les conclusions pouvant être tirées des tests. En effet, ces instances ne reflètent pas forcément la réalité des problèmes auxquels sont confrontées des firmes de construction au niveau de la planification tactique. Les nombres considérés des ressources, des lots de travaux, ainsi que des couples pouvant chevaucher peuvent être différents en pratique. De plus, les données temporelles, ainsi que celles sur les antériorités, les charges requises, les retouches, les quantités disponibles et les coûts des ressources ne se basent pas sur des projets réels de construction.
- Les objectifs possibles dans notre modèle sont la date de livraison du projet, son

coût, ou une combinaison des deux. Or en réalité, la planification en entreprise de construction peut avoir de multiples autres considérations non envisagées dans notre modèle (par exemple financiers, climatiques, environnementaux).

— L'approche par programmation linéaire en nombres entiers limite la résolution en temps raisonnable à des instances relativement simples. Pour des instances de plus grandes tailles, les temps de calcul peuvent exponentiellement augmenter rendant l'approche exacte inutilisable.

Compte tenu des limitations observées, plusieurs voies futures d'amélioration sont proposées dans la section suivante.

5.2 Voies d'amélioration

Afin d'estimer les quantités de retouches, l'exploitation de données historiques pour des projets semblables pourrait être envisagée, mais se heurte elle aussi à d'autres limitations. D'une part, il est peu probable que tous les modes de chevauchement possibles aient été suffisamment testés dans le passé et d'autre part, il est très difficile d'établir une corrélation directe entre les charges supplémentaires et le chevauchement. En effet, des charges additionnelles observées sur des couples qui chevauchent peuvent ne pas être causées par le chevauchement, mais par d'autres facteurs (par exemple accidents ou erreurs). Une solution possible serait de considérer les retouches différemment, non pas en tant que données connues, mais comme étant le résultat d'une interaction plus complexe entre deux activités qui chevauchent.

De plus, les relations d'antériorité peuvent être généralisées afin de couvrir de nouvelles possibilités de chevauchement. Ensuite, il serait plus réaliste de proposer un modèle stochastique prenant en compte les risques d'avoir des aléas dans le projet.

En outre, même si cela paraît être un travail fastidieux et sujet à des limitations de confidentialité, il serait intéressant de tester le modèle sur des instances issues de vrais projets de construction afin de pouvoir tirer des conclusions plus pertinentes pour ce domaine. Le modèle pourrait servir à tester rapidement un grand nombre d'instances de projets, ce qui permettrait d'effectuer des études statistiques pour comprendre les liens entre les décisions optimales de chevauchement et les caractéristiques des projets. Ainsi, il serait possible de tirer des conclusions générales, destinées aux praticiens, pour les guider dans leurs décisions de chevauchement.

Finalement, le développement d'approches plus performantes pour les grandes instances semble être nécessaire pour pouvoir utiliser le modèle en pratique sur de vrais projets. Une voie possible de recherche est la création d'une heuristique à partir du modèle proposé. La

section suivante propose une telle heuristique et présente les résultats préliminaires obtenus.

5.3 Solution alternative

Pour les instances les plus difficiles, la résolution exacte peut nécessiter un temps de calcul très long avant de trouver une solution réalisable. Dans ces cas, il serait difficilement envisageable d'avoir une solution acceptable dans des temps de calcul raisonnables. Nous avons donc exploité notre modèle exact pour servir de base à une heuristique, dont le principe est de trouver rapidement une solution admissible et d'essayer de l'améliorer à chaque itération. Le fonctionnement complet de cette heuristique est expliqué dans la Figure 5.1.

L'heuristique résout d'abord le problème sans contraintes de ressources, en minimisant la date de fin du projet. Ce problème, sans aucune contrainte de ressource, réussit à avoir une solution optimale très rapidement.

Ensuite, l'heuristique cherche à améliorer cette solution initiale au cours d'un nombre prédéfini d'itérations. À chaque itération, l'heuristique parcourt tous les lots, en les sélectionnant un par un. Pour chaque lot, toutes les variables binaires z_{ip}^{c} sont fixées aux valeurs obtenues de la solution courante, à part celles qui concernent le lot sélectionné. Ensuite le problème avec contraintes de ressources est résolu, en minimisant un objectif lié au coût. Ainsi, à chaque résolution, les périodes de début et de fin de toutes les parties des lots sont fixées, à part celles d'un seul lot, ce qui réduit considérablement la combinatoire.

Par ailleurs, pour toutes les instances, les réseaux d'antériorité sont acycliques et les lots sont numérotés avec un ordre topologique (un prédécesseur a toujours un numéro inférieur à ses successeurs). Les lots sont parcourus en sens inverse de leur numéro pour permettre une meilleure diversification. En effet, les lots dans la solution initiale sont tassés vers le début du projet. Il est alors plus efficace de replanifier les successeurs avant les prédécesseurs pour sortir des minimas locaux.

De plus, le terme $(tf_j \cdot \frac{NbIterations-k}{NbIterations})$ permet une plus grande diversification dans l'heuristique en encourageant le lot "libre" à avoir une date de fin plus grande. À chaque itération, le coefficient $\frac{NbIterations-k}{NbIterations}$ diminue, jusqu'à s'annuler lors de la dernière itération ($k = NbIterations$) où l'objectif devient simplement de minimiser le coût.

Deux comportements extrêmes peuvent être distingués en comparant les résultats obtenus par l'heuristique avec ceux de la méthode exacte en fonction du temps. Ces deux cas sont présentés sur deux exemples dans la Figure 5.2. Pour certaines instances, la méthode exacte trouve la valeur optimale en peu de temps, alors que l'heuristique trouve une solution non nécessairement optimale en plus de temps. Pour d'autres instances, l'heuristique trouve une

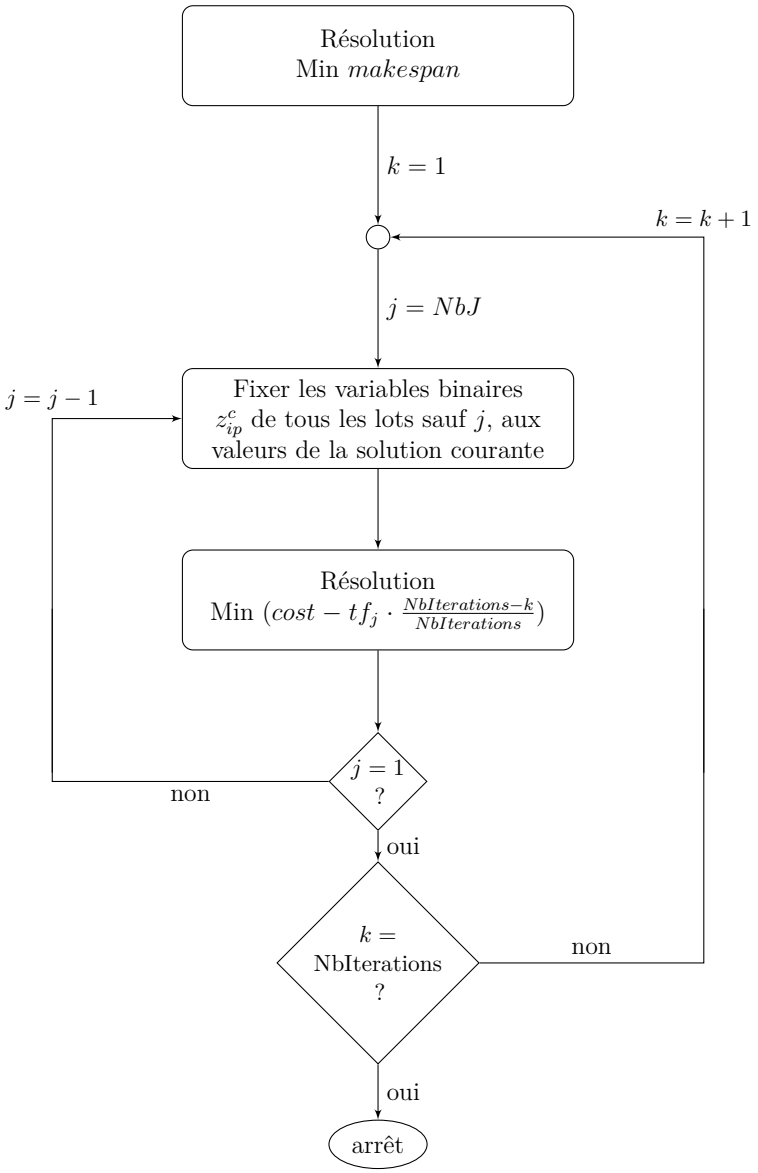

Figure 5.1 Principe de fonctionnement de l'heuristique

solution réalisable rapidement, alors que la solution exacte en trouve possiblement une après de longs temps de calcul.

Les performances de l'heuristique sont comparées à celles de la méthode exacte pour trois temps limites dans le Tableau 5.1. Les deux premières lignes présentent la moyenne et le maximum des temps de calcul. La troisième ligne fournit le pourcentage des instances pour lesquelles aucune solution réalisable n'a été trouvée. La quatrième ligne présente le pourcentage des instances pour lesquelles la solution de l'heuristique a été meilleure que la méthode exacte. Finalement, la dernière ligne donne l'écart relatif moyen des solutions obtenues par

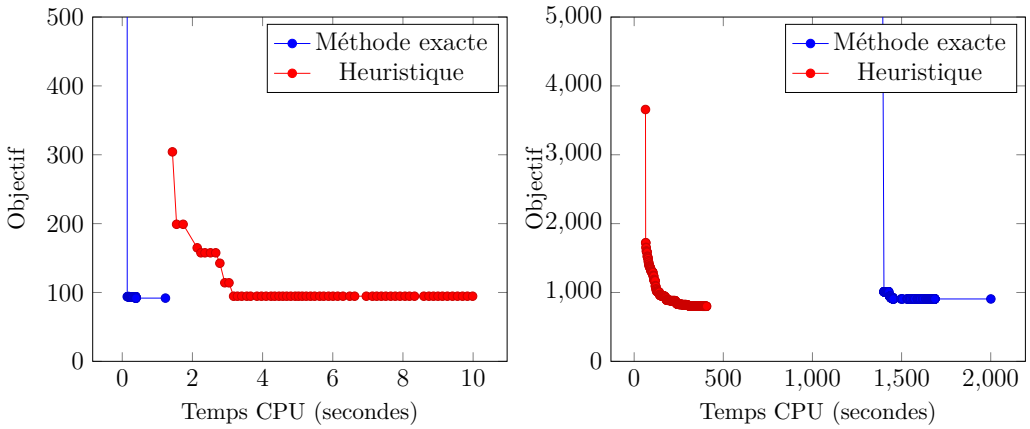

Figure 5.2 Deux comportements extrêmes obtenus par comparaison de l'heuristique à la méthode exacte

l'heuristique par rapport à celles obtenues par la méthode exacte. Cette heuristique simple permet d'abord de trouver des solutions réalisables pour toutes les instances. De plus, elle trouve dans certains cas de meilleures bornes supérieures que la méthode exacte en des temps de calcul inférieurs en moyenne.

5.4 Implication pratique du modèle

Malgré ses limitations, la solution exacte proposée reste pertinente et ses implications pratiques importantes.

Le modèle sert d'outil de grand intérêt aux praticiens pour les aider dans la planification lors des phases préliminaires des projets. Les outils existants ne considèrent pas les possibilités de chevauchement, ce qui oblige les planificateurs à se baser sur leur expérience et leur intuition pour décider des chevauchements des lots de travaux dans un projet. La solution proposée leur permet ainsi de prendre des décisions plus pertinentes et réfléchies concernant les chevauchements.

De plus, le modèle permet une planification selon plusieurs objectifs possibles. Ainsi, il est possible de tester rapidement plusieurs solutions avec divers objectifs, pour permettre aux gestionnaires de choisir la solution qui leur convient le mieux.

Par ailleurs, dans un cadre de planification hiérarchique, le modèle permet aux planificateurs de déterminer les dates de début et de fin des lots, ainsi que les quantités requises de ressources par périodes. Ces informations trouvées au niveau tactique seraient alors utilisées comme contraintes pour le calendrier des activités au niveau opérationnel (Kis and Kovács, 2012).

Tableau 5.1 Performances de l'heuristique comparées à celles de la méthode exacte pour trois limites de temps

	Heuristique	Méthode exacte $T_{limite} = 500s$	Méthode exacte $T_{limite} = 1000s$	Méthode exacte $T_{limite} = 5000s$
Moyenne temps CPU (en s)	191	244	451	1900
Maximum temps CPU (en s)	878	500	1000	5000
% intances/pas de solution trouvée	0.00%	5.28%	3.33%	0.84%
% intances/sol. heursitique est meilleure	-	14.58%	12.98%	6.4%
Écart relatif sol. heuristique/exacte	-	10.90%	16.44%	20.86%

5.5 Conclusion

Malgré nos efforts de proposer un modèle réaliste, notre approche possède ses limites. Ce chapitre a permis de mettre en perspective les limitations du modèle, de proposer des voies d'amélioration permettant de repousser certaines limites et de discuter des implications pratiques du modèle.

CHAPITRE 6 CONCLUSION

Ce mémoire a porté sur la planification tactique de projet avec possibilités de chevauchement des lots de travaux. Le problème provient du besoin des grandes firmes de construction, qui utilisent couramment des techniques d'ingénierie parallèle, telles que le chevauchement, sans disposer d'outils de planification tactique adéquats. Les praticiens produisent alors des échéanciers en se basant sur leur expérience pour décider des niveaux de chevauchement, sans avoir de méthode rigoureuse permettant de quantifier les effets de ces décisions sur les performances globales du projet.

La littérature scientifique présente une lacune dans ce domaine, puisqu'aucun modèle de planification agrégée n'autorise de manière satisfaisante le chevauchement. En effet, malgré les travaux de plusieurs auteurs proposant des liens d'antériorité généralisés pour la planification agrégée, aucun modèle de planification tactique ne traite à proprement dit du chevauchement. Par ailleurs, plusieurs modèles de chevauchement ont été proposés pour la planification opérationnelle, mais ne peuvent pas être utilisés au niveau tactique.

Pour répondre à ce besoin, nous avons proposé un modèle mathématique de programmation linéaire en nombres entiers dédié au problème RCCP avec chevauchement. Ce modèle considère plusieurs modes de chevauchement entre des lots de travaux. Un mode est caractérisé par le pourcentage du prédécesseur (en termes de charge de travail) devant être complété avant de pouvoir commencer le successeur et les quantités correspondantes de retouches sur le successeur et le prédécesseur. Le modèle a la particularité de combiner des représentations continue et discrète du temps. En effet, l'horizon du temps est divisé en plusieurs périodes sur lesquelles les consommations des ressources sont évaluées, alors que les dates de début et de fin, ainsi que les évènements d'atteinte de pourcentages de complétion d'un lot, sont continues et peuvent se produire à l'intérieur de périodes.

Le modèle a été implanté et testé sur 5 ensembles de 450 instances générées par modification d'instances existantes. Plus que la moitié de ces instances a été résolue à l'optimum en moins de 500 secondes, alors que la recherche s'est arrêtée pour moins que le tiers des instances avant l'atteinte de la solution optimale. Une analyse des temps de résolution suggère que la considération du chevauchement sur la performance du modèle est néfaste, mais a moins d'impact que le *slack* et le nombre de lots. Par ailleurs, l'analyse de solutions a montré que le chevauchement améliore aussi bien le coût de projet que sa durée.

Finalement, compte tenu des longs temps de résolution observés pour certaines instances, une heuristique qui se base sur le modèle mathématique a été développée. Cette solution

alternative, en dépit de sa simplicité, semble donner des résultats encourageants en repoussant certaines limites de la résolution exacte. La voie de recherche consistant à exploiter le modèle exact pour développer des heuristiques semble ainsi être pertinente.

Pour récapituler, les contributions de notre recherche se situent à trois niveaux. D'abord, nous avons proposé un modèle mathématique original qui répond à une problématique réelle. Ensuite, nous avons utilisé ce modèle pour analyser les effets du chevauchement sur les coûts et les durées de projets. Finalement, nous avons proposé une solution alternative sous forme de matheuristique.

Malgré nos efforts de proposer un modèle réaliste, notre approche possède certaines limitations, concernant surtout son applicabilité à la pratique. En effet, les données requises sur les quantités de retouches nécessaires à chaque mode sont difficilement accessibles en pratique. Des limitations concernant le caractère aléatoire de plusieurs données, les fonctions objectifs proposées et les temps de calcul requis ont également été soulevées.

En dépit de ces limites, notre étude reste tout de même pertinente. En effet, le modèle proposé est original dans le sens où, à notre connaissance, aucun autre modèle ne propose une planification au niveau agrégé autorisant le chevauchement des lots de travaux. Ensuite, le problème a été défini de manière réaliste puisqu'il est fortement inspiré de la pratique des compagnies de construction. De plus, l'approche exacte retenue permet d'obtenir des solutions optimales et donc de servir de référence pour évaluer la qualité des solutions de futures heuristiques. Le modèle peut également être à la base de matheuristiques.

Compte tenu des limitations soulevées, plusieurs voies d'améliorations ont été proposées. Elles concernent essentiellement la considération des quantités de retouches en tant que sorties du problème, la prise en compte de l'incertitude, l'essai du modèle sur de vrais projets et le développement d'heuristiques plus performantes sur de grandes instances.

Pour conclure, la solution proposée et les résultats obtenus sont encourageants, mais de futures améliorations semblent être nécessaires pour que l'approche soit directement applicable en pratique. Notre travail de recherche va tout de même dans la direction d'avoir des méthodes de planification plus réalistes, offrant la possibilité de considérer des techniques d'ingénierie parallèle couramment employées par les praticiens.

RÉFÉRENCES

A. Alfieri, T. Tolio, et M. Urgo, "A project scheduling approach to production planning with feeding precedence relations," *International Journal of Production Research*, vol. 49, no. 4, pp. 995–1020, 2011.

R. N. Anthony, "Planning and control systems : a framework for analysis," Boston, 1965. [En ligne]. Disponible : http://opac.inria.fr/record=b1082278

C. Backhouse et N. Brookes, *Concurrent Engineering : What's Working where*. Design Council, 1996. [En ligne]. Disponible : http://books.google.ca/books?id=QzRhe0MM_EkC

F. Berthaut, P. Robert, P. Nathalie, et H. Adnène, "Time-cost trade-offs in resource-constraint project scheduling problems with overlapping modes," *International Journal of Project Organisation and Management*, vol. 6, no. 3, pp. 215–236, 2014.

L. Bianco et M. Caramia, "Minimizing the completion time of a project under resource constraints and feeding precedence relations : an exact algorithm," *4OR*, vol. 10, no. 4, pp. 361–377, 2012. [En ligne]. Disponible : http://dx.doi.org/10.1007/s10288-012-0205-0

L. Bianco et M. Caramia, "A new formulation for the project scheduling problem under limited resources," *Flexible Services and Manufacturing Journal*, vol. 25, no. 1-2, pp. 6–24, 2013. [En ligne]. Disponible : http://dx.doi.org/10.1007/s10696-011-9127-y

S. M. Bogus, K. R. Molenaar, et J. E. Diekmann, "Concurrent engineering approach to reducing design delivery time," *Journal of construction engineering and management*, vol. 131, no. 11, pp. 1179–1185, 2005.

C.-S. Chen, S. Mestry, P. Damodaran, et C. Wang, "The capacity planning problem in make-to-order enterprises," *Mathematical and Computer Modelling*, vol. 50, no. 9–10, pp. 1461 – 1473, 2009.

K. Cherkaoui, R. Pellerin, P. Baptiste, et N. Perrier, "Planification hiérarchique de projets EPCM," dans *10ème Conférence Internationale de Génie Industriel 2013*, La Rochelle, France, 12–14 Jun. 2013.

R. De Boer, "Resource-constrained multi-project management : a hierarchical decision support system," Thèse de doctorat, University of Twente, Enschede, The Netherlands, Oct. 1998.

R. Dehghan et J. Ruwnapura, "Model of trade-off between overlapping and rework of design activities," *Journal of Construction Engineering and Management*, vol. 140, no. 2, p. 04013043, 2014. [En ligne]. Disponible : http://ascelibrary.org/doi/abs/10.1061/%28ASCE%29CO.1943-7862.0000786

S. E. Elmaghraby et J. Kamburowski, “The analysis of activity networks under generalized precedence relations (gprs),” *Manage. Sci.*, vol. 38, no. 9, pp. 1245–1263, Sep. 1992. [En ligne]. Disponible : http://dx.doi.org/10.1287/mnsc.38.9.1245

N. Gademann et M. Schutten, “Linear-programming-based heuristics for project capacity planning,” *IIE Transactions*, vol. 37, no. 2, pp. 153–165, 2005.

J. E. V. Gerk et R. Y. Qassim, “Project Acceleration via Activity Crashing, Overlapping, and Substitution,” *IEEE Transactions on engineering management*, vol. 55, no. 4, pp. 590–601, Nov. 2008.

L. Grèze, R. Pellerin, P. Leclaire, et N. Perrier, “A heuristic method for resource-constraint project scheduling with activity overlapping,” *Journal of Intelligent Manufacturing*, 2012, forthcoming.

A. Y. Ha et E. L. Porteus, “Optimal timing of reviews in concurrent design for manufacturability,” *Management Science*, vol. 41, no. 9, pp. 1431–1447, 1995.

E. Hans, “Resource loading by Branch-and-Price techniques,” Thèse de doctorat, University of Twente, Enschede, The Netherlands, Oct. 2001.

E. Hans, W. Herroelen, R. Leus, et G. Wullink, “A hierarchical approach to multi-project planning under uncertainty,” *Omega*, vol. 35, no. 5, pp. 563 – 577, 2007. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0305048305001581

A. Haït et G. Baydoun, “A new event-based MILP model for the resource-constrained project scheduling problem with variable intensity activities,” dans *The IEEE International Conference on Industrial Engineering and Engineering Management 2012*, Honk Kong, 10–13 Déc. 2012.

W. Herroelen, “Project scheduling—theory and practice,” *Production and Operations Management*, vol. 14, no. 4, pp. 413–432, 2005. [En ligne]. Disponible : http://dx.doi.org/10.1111/j.1937-5956.2005.tb00230.x

H. Kerzner, *Project Management : A Systems Approach to Planning, Scheduling, and Controlling*. John Wiley & Sons., 2001.

T. Kis et A. Kovács, “A cutting plane approach for integrated planning and scheduling,” *Computers and Operations Research*, vol. 39, no. 2, pp. 320 – 327, 2012. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0305054811001079

T. Kis, “A branch-and-cut algorithm for scheduling of projects with variable-intensity activities,” *Mathematical programming*, vol. 103, no. 3, pp. 515–539, 2005.

T. Kis, “RCPS with variable intensity activities and feeding precedence constraints,” dans *Perspectives in modern project scheduling*, J. Józefowska et J. Weglarz, éds. Berlin : Springer, 2006, ch. 5, pp. 105–129.

R. Kolisch, A. Sprecher, et A. Drexl, “Characterization and generation of a general class of resource-constrained project scheduling problems,” *Management science*, vol. 41, no. 10, pp. 1693–1703, 1995.

X. Koufteros, M. Vonderembse, et W. Doll, “Concurrent engineering and its consequences,” *Journal of Operations Management*, vol. 19, no. 1, pp. 97 – 115, 2001. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0272696300000486

V. Krishnan, S. D. Eppinger, et D. E. Whitney, “A model-based framework to overlap product development activities,” *Manage. Sci.*, vol. 43, no. 4, pp. 437–451, Avr. 1997. [En ligne]. Disponible : http://dx.doi.org/10.1287/mnsc.43.4.437

J. Lin, Y. Qian, W. Cui, et Z. Miao, “Overlapping and communication policies in product development,” *European Journal of Operational Research*, vol. 201, no. 3, pp. 737 – 750, 2010. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/S0377221709002008

C. H. Loch et C. Terwiesch, “Communication and uncertainty in concurrent engineering,” *Management Science*, vol. 44, no. 8, pp. 1032–1048, 1998. [En ligne]. Disponible : http://pubsonline.informs.org/doi/abs/10.1287/mnsc.44.8.1032

M. Masmoudi, E. W. Hans, R. Leus, et A. Haït, “Fuzzy multi-project rough-cut capacity planning,” dans *Sequencing and Scheduling with Inaccurate Data*, Y. N. Sotskov et F. Werner, éds. Nova Science Publishers, 2014, ch. 1.

R. H. Möhring, “Minimizing costs of resource requirements in project networks subject to a fixed completion time,” *Operations Research*, vol. 32, no. 1, pp. 89–120, 1984. [En ligne]. Disponible : http://dx.doi.org/10.1287/opre.32.1.89

A. Naber et R. Kolisch, “MIP models for resource-constrained project scheduling with flexible resource profiles,” *European Journal of Operational Research*, vol. 239, no. 2, pp. 335–348, 2014. [En ligne]. Disponible : http://dx.doi.org/10.1016/j.ejor.2014.05.036

S. Nicoletti et F. Nicolo, “A concurrent engineering decision model : Management of the project activities information flows,” *International Journal of Production Economics*, vol. 54, no. 2, pp. 115–127, 1998.

F. Peña-Mora et M. Li, “Dynamic planning and control methodology for design/build fast-track construction projects,” *Journal of Construction Engineering and Management*, vol. 127, no. 1, pp. 1–17, 2001. [En ligne]. Disponible : http://ascelibrary.org/doi/abs/10.1061/%28ASCE%290733-9364%282001%29127%3A1%281%29

B. Prasad, *Concurrent Engineering Fundamentals, Volume II : Integrated Product Development*, sér. Concurrent Engineering Fundamentals. Prentice Hall PTR, 1996. [En ligne]. Disponible : http://books.google.ca/books?id=tyofAQAAIAAJ

A. Pritsker et L. Watters, *A Zero-one Programming Approach to Scheduling with Limited Resources*, sér. Memorandum (Rand Corporation) RM-5561-PR. Rand Corporation, 1968.

T. A. Roemer et R. Ahmadi, "Concurrent Crashing and Overlapping in Product Development," *Operations Research*, vol. 52, no. 4, pp. 606–622, 2004.

T. A. Roemer, R. Ahmadi, et R. H. Wang, "Time-cost tradeoffs in overlapped product development," *Oper. Res*, pp. 858–865, 2000.

F. N. Talla, R. Leus, K. Nip, et Z. Wang, "Resource loading with time windows," 2014. [En ligne]. Disponible : http://ssrn.com/abstract=2384455

J. Weglarz, "Project scheduling with continuously-divisible, doubly constrained resources," *Management Science (pre-1986)*, vol. 27, no. 9, pp. 1040–1053, 09 1981.

R. Winner, J. P. Pennel, H. E. Bertrand, et M. M. G. Slusarczuk, *The Role of Concurrent Engineering in Weapons System Acquisition*, sér. IDA Report R-338. Alexandria, Virginia, USA : Institute for Defense Analysis, 1988.

G. Wullink, A. J. R. M. Gademann, E. W. Hans, et A. van Harten, "Scenario-based approach for flexible resource loading under uncertainty," *International Journal of Production Research*, vol. 42, no. 24, pp. 5079–5098, 2004. [En ligne]. Disponible : http://www.tandfonline.com/doi/abs/10.1080/00207540001733887

G. Wullink, "Resource loading under uncertainty," Thèse de doctorat, University of Twente, Enschede, The Netherlands, Mar. 2005.